DOCUMENT RESUME

ED 055 252		VT 014 134
AUTHOR	Magisos, Joel H.; Berkebile, Lonnie A.	
TITLE	Computer Applications to ERIC Clearinghouse Document Collections: A Final Report.	
INSTITUTION	Ohio State Univ., Columbus. Center for Vocational and Technical Education.	
PUB DATE	Mar 71	
NOTE	323p.	
EDRS PRICE	MF-$0.65 HC-$13.16	
DESCRIPTORS	*Computer Programs; Data Bases; Information Centers; Information Networks; *Information Processing; *Information Retrieval; Information Systems; *Information Utilization; Manuals; *Program Development; Research Tools; Research Utilization; Vocational Education	
IDENTIFIERS	ERIC Clearinghouse Document Collections	

ABSTRACT

The purpose of this project was to develop computer techniques for the production of VT-ERIC publications and to report these techniques so that they might be used by other ERIC Clearinghouses for production of their publications. The Clearinghouse needed to: (1) generate bibliographies from its own collection upon which to base information analysis papers, (2) conduct searches of the collection for solution to problems in the field, (3) generate camera-ready copy for publication of Clearinghouse abstract journals, and (4) describe the system to facilitate its installation in Clearinghouses and other agencies. Included in the report are an explanation of the system requirements, a description of computer program operating procedures, capability, and utilization, and a summary and recommendations. Extensive appendixes provide the technical details for the system including keypunching, job control language, control cards, error messages, program listing, program and system flow charts, and page samples. A related document, Computer Applications to ERIC Clearinghouse Document Collections: A User and Installation Handbook is available as VT 014 135. (Author/JS)

ED055252

VT014134

ERIC CLEARINGHOUSE
THE CENTER FOR VOCATIONAL
AND TECHNICAL EDUCATION
THE OHIO STATE UNIVERSITY
COLUMBUS, OHIO

ED055252

U.S. DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
OFFICE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

COMPUTER APPLICATIONS TO ERIC CLEARINGHOUSE DOCUMENT COLLECTIONS: A FINAL REPORT.

by

Joel H. Magisos and Lonnie A. Berkebile

ERIC Clearinghouse on Vocational and Technical Education
The Center for Vocational and Technical Education
The Ohio State University
Columbus, Ohio 43210

March 1971

FG poor

PREFACE

The ERIC Clearinghouse on Vocational and Technical Education (VT-ERIC) has been producing two quarterly abstract journals since late 1967. Abstracts of Research and Related Materials in Vocational and Technical Education (ARM) provides indexes to and abstracts of documents in vocational and technical education; it is designed for researchers, teacher educators, administrators, and others who need research and related materials in an organized and accessible form. Abstracts of Instructional Materials in Vocational and Technical Education (AIM) provides indexes to and abstracts of instructional materials in vocational and technical education; it is designed to provide accessibility to the collection for teachers, administrators, curriculum specialists, and others. Nearly 36 percent of the materials in ARM and 79 percent of those in AIM have not appeared in Research in Education and are made available only in this way to over 2,800 users who receive the publications.

The application of computer technology to the generation of AIM and ARM has afforded the Clearinghouse with economies and efficiencies otherwise not possible and has opened up new opportunities to utilize the Clearinghouse collection such as document retrieval and information product repackaging. Other Clearinghouses may find these techniques applicable to their operation. Agencies, such as state vocational-technical education research coordination units, will soon be able to acquire the ARM and AIM collection on magnetic tape and be able to retrieve document resumes with QUERY. VT-ERIC will continue to use and update the files.

Recognition should be given to Joel H. Magisos, coordinator of information services at The Center, and Lonnie A. Berkebile, computer specialist at The Center, for their work in planning, conducting, and reporting the project.

Robert E. Taylor
Director
ERIC Clearinghouse on Vocational
and Technical Education

INTRODUCTION

The project reported herein was conducted by the ERIC Clearinghouse on Vocational and Technical Education pursuant to an extension of its contract. The purpose of the project was to develop computer techniques for the production of *Abstracts of Research and Related Materials in Vocational and Technical Education* (ARM) and *Abstracts of Instructional Materials in Vocational and Technical Education* (AIM) and to report these techniques so that they might be used by other ERIC Clearinghouses for production of Clearinghouse publications. The specific objectives of the project were to enable the Clearinghouse to:

1. generate bibliographies from the Clearinghouse collection upon which to base information analysis papers,
2. conduct searches of the Clearinghouse collection for solution to problems in the field,
3. generate camera-ready copy for publication of Clearinghouse abstract journals (i.e., ARM, AIM, and AIM-ARM Annual Index), and
4. describe the system to facilitate its installation in Clearinghouses and other agencies.

Figure 1 on page vi illustrates a general outline of the system. A schematic flowchart of the programs is provided in Figure 2, page 18 - 21.

The document collection of the ERIC Clearinghouse on Vocational and Technical Education has been made available through announcement in USOE's monthly Research in Education (RIE) and through its own quarterly abstract series, Abstracts of Research and Related Materials in Vocational and Technical Education (ARM) and Abstracts of Instructional Materials in Vocational and Technical Education (AIM).

Efficiency and accuracy of ARM and AIM production has been gained through the application of computer techniques. Bibliographic data and abstracts are brought together on data cards for input to magnetic tape files. The magnetic tape file provides storage of the data. The computer programs developed in this project permit accurate manipulation of the data into formats for resume and index sections of AIM and ARM which may be printed out as camera-ready copy. Production of the Fall 1970 issue of ARM represented the first Clearinghouse publication which had been completely computer-generated. In addition, the Fall 1970 ARM contained the Clearinghouse's backlog and was released on schedule.

In the first twelve issues of ARM, 3,703 documents were announced, 1,324 of which had not been reported in RIE. In the first twelve issues of AIM, 2,605 documents were announced, of which 2,067 had not been reported in RIE. Thus, 3,391 of the documents announced in AIM and ARM could not be searched by computer until after the development of the software package reported here. The first search of a partial file has been successfully conducted and input of resumes in back issues of AIM and ARM to magnetic tape is nearly complete. A significant by-product of the project will be the eventual availability of the AIM and ARM collection on magnetic tape in a format to permit others to search it with QUERY.

The report includes explanation of the system requirements; descriptions of computer program operating procedures, capability, and utilization; and a summary and recommendations. Extensive appendices provide the technical details for the system including keypunching, job control language, control cards, error messages, program listings, program and system flowcharts, and page samples. Also available is Computer Applications to ERIC Clearinghouse Document Collections: A User and Installation Handbook by Lonnie A. Berkebile.

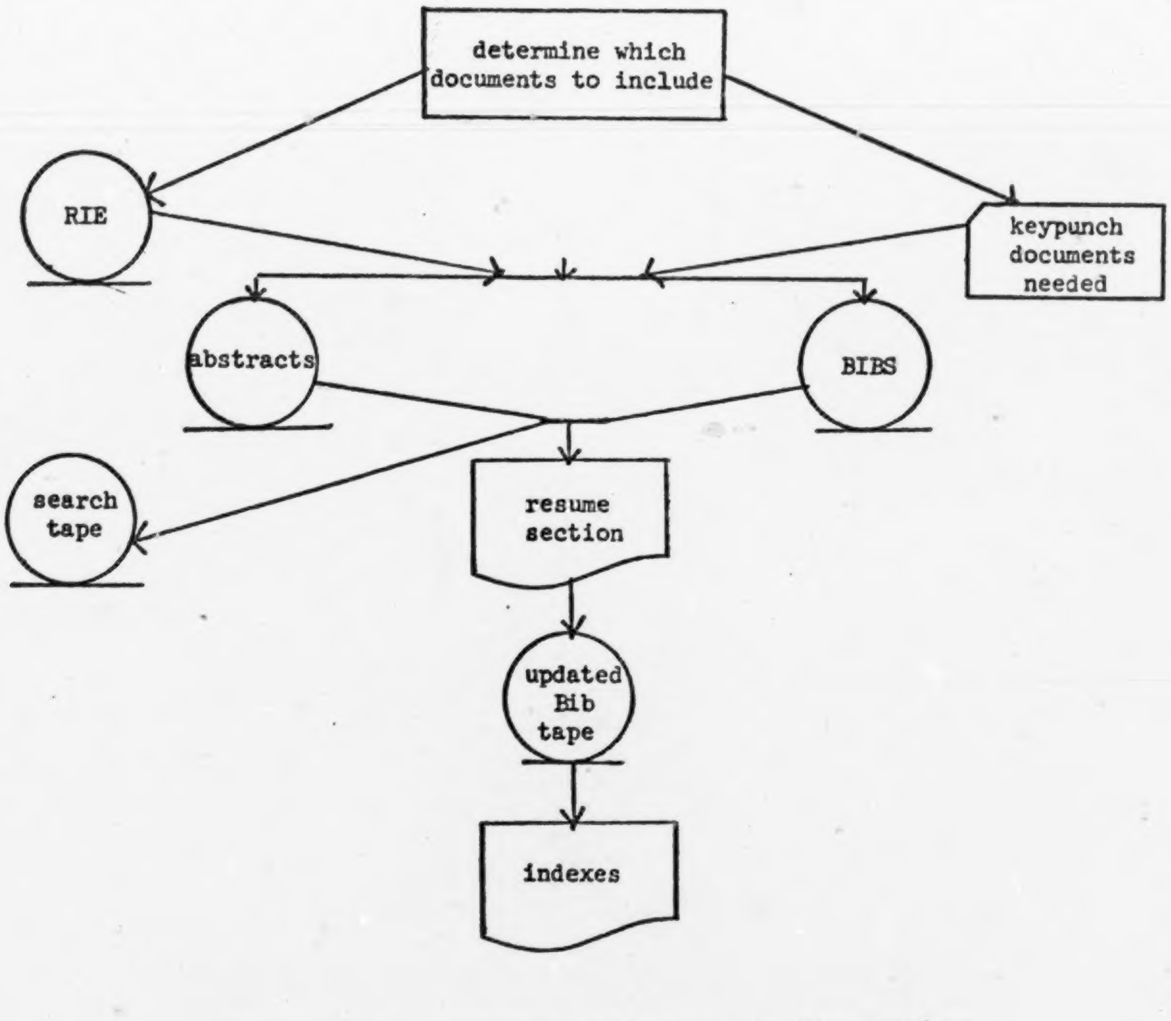

determine which documents to include
RIE
keypunch documents needed
abstracts
BIBS
search tape
resume section
updated Bib tape
indexes

Figure 1 -- General Outline of the System

TABLE OF CONTENTS

PREFACE ii

INTRODUCTION iv

TABLE OF CONTENTS vii

LIST OF FIGURES viii

Chapter

I. SYSTEM REQUIREMENTS FOR EFFECTIVE COMPUTER UTILIZATION 1

Equipment Requirements for Clearinghouse Computer Programs
Storage of Data Pertaining to Clearinghouse Documents
Bibliographic Record Format
Production of Clearinghouse Information Products
Retrieval of Clearinghouse Documents

II. DESCRIPTION OF COMPUTER PROGRAM OPERATING PROCEDURES, CAPABILITY, AND UTILIZATION IN ERIC CLEARINGHOUSES 8

Indexes
The Document Number Index
Supporting Services Index
Subject Index
Author Index
Conversion of Document Number Index
Edit Programs
Master Edit Programs
Annual Subset Edit Programs
Use of QUERY Program

III. SUMMARY AND RECOMMENDATIONS 16

Summary of the Project
Recommendations

APPENDICES 1

LIST OF FIGURES

Figure page

1. General Outline of the System vi

2. Schematic Flow Chart of Computer Programs 18

SYSTEM REQUIREMENTS FOR EFFECTIVE COMPUTER UTILIZATION

The computer software that was developed in this project permits effective utilization of the computer in the production of Clearinghouse abstract journals such Abstracts of Research and Related Materials in Vocational and Technical Education (ARM) and Abstracts of Instructional Materials in Vocational and Technical Education (AIM). It also permits computer search of the document collection which is announced in the abstract journal. The principal benefits are increase in production efficiency and accuracy, access to the Clearinghouse document collection by computer, and increase in the options available for repackaging the document collection in other useful forms.

The following sections briefly describe the underlying requirements for, and general nature of, the procedures for data storage, publication production, and search tape preparation.

The first task was to prepare a package which could easily be converted to other computing operations and still conserve time in the program operation. Possible languages to be used were FORTRAN, ASSEMBLER, or PL/I. The first two languages have requirements which should be avoided. FORTRAN has limited character manipulation and presents difficulties with variable length records. ASSEMBLER would be difficult to modify and update. Thus, a decision was made to write the programs in PL/I, even though the package would be restricted to IBM installations. It might be noted, however, that other manufacturers are now producing PL/I compilers.

Equipment Requirements for Clearinghouse Computer Programs

The programs in this package have been run on an IBM 360 model 75 operating under HASP-II. All of the programs in this package will fit in 126,000 bytes of main memory. The running times that we have experienced indicate that 2 hours of computer time for each publication will be needed. The 2 hour estimate is for a publication of 300 documents.

The equipment needed for producing an abstract journal and magnetic search tape is available at most computer centers with an IBM 360 computer. If both upper and lower case letters are used, each document will require approximately 30 minutes to keypunch on an IBM 029 keypunch. In addition to a keypunch for preparing documents for publication, a PL/I compiler, 3 tape drives, a card reader and line printer will be required. In addition all of the index programs require access to the SORT/MERGE routines. The access to SORT/MERGE will be more efficient if disk storage is available; however if sufficient tape drives are available, the sorting can be handled on tapes.

Storage of Data Pertaining to Clearinghouse Documents

Storage and maintenance of data is a first problem when preparing to create a publication. What format should be used for the data? What storage device should be used? These are only two of the questions to be answered. One alternative is storage on disk providing a direct access system which would be very efficient. The problem with this approach is the storage costs, which are relatively high, thus magnetic tape storage was advisable. Data must be prepared for input on available equipment and, in this case, it amounted to a choice between paper tape and punched cards.

Because of available equipment, paper tape would have to be converted to punched cards as an intermediate step to magnetic tape, thus initial preparation of cards was a logical decision. If the hardware available had allowed direct conversion from paper tape to magnetic tape, the choice to use paper tape would have been feasible. Temporary retention of data cards affords security for the data; but, after printing of the publication, these cards may be destroyed.

The bibliographic data in ERIC resumes varies both in length and in types of data available. Originally, the Clearinghouse format was variable and differed from the format being used by Central ERIC . This variable format was the basis for card preparation and magnetic tape file creation. The program to convert the card data to magnetic tape is referred to as the MASTER program.

The MASTER program reads in each card for a particular document and analyzes the card for syntax. This program pulls off all card images related to a document by using the accession number. The accession number is in the first ten positions of the record. Then the program searches the card images to discover what data is applicable and what is or is not included. The program then sets up one record on the MASTER tape for the document. The length of the record varies due to the varying material, and there is a record for every document in the quarterly abstract series (ie., AIM and ARM). This record is set up so that all bibliographic data is listed first in a special order, followed by a section of numbers that gives the lengths of the different data. A zero length implies that a data piece is missing from the record. The lengths and zeroes allow access to any particular piece of bibliographic information within this large record, and the zero lengths indicate certain bibliographic information is missing. The order and format of the data in each record is as follows: (Asterisks indicate material which does not always appear.)

Bibliographic Record Format

	Piece of Data	Length (in bytes)
1.	Document Identification Number (VT Number)	10
2.	Code Number	17
3.	Accession Number (ED Number)	20
4.	Whether it is an AIM or ARM publication and the current season	3
5.	Current Year	4
6.	Page number on which the resume appears in the front section of publication	4
7.	*Author$_1$, . . . Author$_n$, $n \leq 2$	≤ 132
8.	Title	≤ 396
9.	*Corporate Author$_1$, . . . Author$_n$, $n \leq 5$	≤ 660
10.	*Sponsoring Agency$_1$, . . . Sp. Agency$_n$, $n \leq 3$	≤ 396
11.	Price, Contract Number, Grant Number, and Program Area Number	≤ 132
12.	*Report Number, Bureau Number	≤ 132
13.	*Availability	≤ 264
14.	*Journal Citation	≤ 132
15.	Publication Date	7
16.	The number of pages in the report	≤ 16

17.	*Descriptive Note	≤ 247
18.	Descriptors	660
19.	*Identifiers	≤ 396
20.	// (separator)	2
21.	Number of authors in Section 7, $0 \le n \le 2$	4
22.	*Length of each author in Section 7 (If n=0 then there is no Section 22.)	4*n
23.	Length of the title	4
24.	Number of corporate authors in Section 9 $0 \le n \le 5$	4
25.	*Length of each corporate author in Section 9 (If n=0, there is no Section 25.)	4*n
26.	Number of sponsoring agencies in Section 10 $0 \le n \le 3$	4
27.	*Length of each sponsoring agency in Section 10 (If n=0, there is no Section 27.)	4*n
28.	Length of the Price	4
29.	Length of the Contract Number	44
30.	Length of the Grant Number	4
31.	Length of the PA Number	4
32.	Length of the Report Number	4
33.	Length of the Bureau Number	4
34.	Length of the Availability	4
35.	Length of the Journal Citation	4
36.	Length of the Publication Date	4
37.	Length of the Page Count	4
38.	Lenth of the Descriptive Note	4
39.	Length of the Descriptors	4
40.	Length of the Identifiers	4

The maximum length of such a record is 3,732 bytes.

Many of the Clearinghouse products of the tape to be produced by this package have been announced in RIE at an earlier date. In the case of these publications, the data is already in machine readable form. The problem of needing a separate format for the publication tapes is easily solved by a program to convert the tapes to the desired Clearinghouse format. Since only bibliographic data is needed for index sections of the publications, the program designed to convert the RIE magnetic tape data creates two new magnetic tapes: one tape for bibliographic data and the second one for abstracts for the documents.

In order to make the publications more useful than a simple bibliography, it is necessary to add abstracts to the finished product. Since the abstract is only used for the resume section of the publications, it is more efficient to have the abstracts on a separate tape.

The format used for storing abstract data is very simple: 10 bytes for the Clearinghouse accession number, 2 bytes to identify the subject area, and 1,988 bytes for the abstract. The 2-byte code used in the abstract data appears in a fixed position in the bibliographic data. The Clearinghouse accession number and the code is both bibliographic and abstract data; thus, the documents can be sorted into several different orders and still allow a program to read both tapes and find corresponding document records.

Production of Clearinghouse Information Products

The first step is to determine which documents are to appear in the Clearinghouse publication. After the documents have been selected, the location of the bibliographic and abstract data must be determined. If the information has appeared in RIE and the quarterly update of ERIC tapes, the document information will not need to be keypunched.

Once the document information is located, bibliographic and abstract tapes need to be created. Once these magnetic tapes have been created, a resume section containing abstracts and bibliographic data can be printed. Then, author, subject, supporting service, and title (or document number) indexes may be printed.

After the resume and indexes are proofed, the camera-ready printout can be prepared and then processed by a printing contractor. This involves reduction of the size and printing of the publication.

Retrieval of Clearinghouse Documents

The bibliographic and abstract tapes can be combined into the ERIC tape format to permit use of QUERY. QUERY allows the capability to search the data base.

II

DESCRIPTION OF COMPUTER PROGRAM OPERATING PROCEDURES, CAPABILITY, AND UTILIZATION IN ERIC CLEARINGHOUSES

Procedures for creation of document number, author, supporting services, subject, and document number conversion indexes are described in the following sections. Further detail on specific programs is provided in the appendixes referred to in the text. Also described are edit programs for bibliographic records, annual document number index data, and all other annual index data. Use of the QUERY programs on Clearinghouse magnetic tapes is explained.

Indexes

Development of the indexes is dependent on the existence of the bibliographic and abstract magnetic tapes that can be created following procedures described in preceding sections of the report. Control data for all indexes are listed in Appendix C. Listings are in Appendix E and flowcharts are in Appendix F.

The Document Number Index

In the document number index, Clearinghouse accession number, ED number, resume section page number, and title for each document are listed. The order of the index is by ascending accession number. In order to produce the document number index, the MASTER file containing the bibliographic information is read by the document number index program. The information needed for the document number index is pulled from the bibliographic tape. Since the year of publication will be needed in the annual index, this is also pulled from the bibliographic tape. The information for each document is stored in its own record on a separate device (i.e., disk or tape). The subset file which is thus created is then sorted by accession number. The records in the subset file consist of:

1. VT number (10 bytes)
2. ED number (20 bytes)
3. Page number (4 bytes)
4. Length of title (4 bytes)
5. Title (maximum of 396 bytes)
6. Issue (3 bytes)
7. Year (4 bytes)

Total record length is ≤ 441 bytes

After these records have been sorted by accession number, the document number index for the quarterly issue is created. This index is first printed and proofed. It is then placed on magnetic tape, and the data on this tape is printed on camera-ready paper. After the index for quarterly issue has been prepared, the annual index can be prepared from this and data sets from other quarterly issues. All of the information from the indexes in the quarterly issue is not used in the annual index. Thus, each record is shortened to contain:

1. Accession number (6 bytes)
2. Page number (4 bytes)
3. Issue (3 bytes)
4. Year (2 bytes)
5. Three blanks to increase length to 18 bytes.

The subset file for the annual index is stored on magnetic tape and this data set is sorted by ascending accession number. The annual document number index contains the shortened records from the document number indexes for the issues which have been published during the year.

Supporting Services Index

To produce the supporting services index, Clearinghouse accession number, ED number, resume section page number, title, and part of the special code (i.e., form, field, and category) is needed. The section of the code needed indicates the field for the document. This code is the 3 characters in the 20-22 column of the code card. For example, in vocational and technical education, these fields might be: agricultural education, business education, distributive education, home economics education, health occupations education, industrial arts education, manpower economics, occupational psychology, occupational sociology, trade and industrial education, and general technical and vocational education. The program is presently set for the fields in vocational and technical education. However, these can easily be adjusted to suit any supportive service required or other categories. The supporting service is not repeated for each document. It is listed once and all the documents having that sub-code are listed under the title. The documents are listed in clearinghouse accession number ascending order under the appropriate supporting service. For each bibliographic record, sub-codes, clearinghouse accession number, ED number, resume section page number, and title are needed for the indexes in the quarterly issues. The publication date will be needed for the annual index. This information is stored on a subset file containing:

1. Sub-code (3 bytes)
2. Clearinghouse accession number (10 bytes)
3. ED number (20 bytes)
4. Publication date (7 bytes)
5. Title (maximum of 396 bytes)

Total record length is $\leq$ 436 bytes.

After the quarterly issue is prepared, the subset files can be shortened further for the annual index. The annual supporting services index, sub-code, publication year, and clearinghouse accession number are the only information needed. The format for the annual supporting services subset file is:

1. Sub-code (3 bytes)
2. Publication year (2 bytes)
3. Clearinghouse accession number (6 bytes)
4. Seven blanks at the end for padding to make record length 18 bytes.

Subject Index

The subject index is made up from starred descriptors and identifiers. The bibliographic tape format is

keyword$_1$; ... keyword$_n$; etc.

Some keywords are starred. These are the terms to be used as index terms. For the subject index, it is necessary to extract all starred identifiers and descriptors, Clearinghouse accession number, ED number, resume section page number, and title for the quarterly subject index, plus the publication year for the annual subject index. For each starred keyword in the record, a record is written on a disk subset file. There are approximately five starred keywords for each record. The records on disk subset files will look as follows:

1. Starred keyword (200 bytes)
2. VT number (10 bytes)
3. ED number (20 bytes)
4. Page number (4 bytes)
5. Publication date (7 bytes)
6. Title ($\leq$ 396 bytes)

Total record length is $\leq$ 637 bytes.

In order to sort according to the starred keywords, the keyword field in the record must be of a fixed length. However, the keywords vary in length, so a maximum keyword length of 200 bytes was selected. When a keyword is not 200 bytes long, it is left justified and then padded with blanks on the right. These records are first sorted by alphabetical order of keywords and then by accession number in ascending order. The subject index is printed and proofed, then copied onto tape to be put on camera-ready paper. Once the subject index has been produced for the quarterly issue, the annual subject subset file can be written. For the annual index, the records will contain:

1. Starred keyword (200 bytes)
2. Publication year (2 bytes)
3. VT number (6 bytes)

Record length is 208 bytes.

Author Index

Each bibliographic record can have personal authors or corporate authors or perhaps neither of the two. The author index uses the author name, accession number, resume section page number, and title for the author index in the quarterly issue, and the publication date for the annual author index. The author index is sorted by authors in alphabetical order. Each author name is listed once in the index and under it is listed all documents written by the author in Clearinghouse accession number ascending order. The record for each author is formed in the author subset file. The records consist of:

1. Author name (132 bytes)
2. Clearinghouse accession number (10 bytes)
3. ED number (20 bytes)
4. Page number (4 bytes)
5. Year (2 bytes)
6. Length of title (3 bytes)
7. Title (maximum of 396 bytes)

Total record length is 567 bytes.

Just as in the subject index, the author name is padded with blanks on the right to enable the sort, the subset file records are sorted by author name in alphabetical order and then by Clearinghouse accession number in ascending order. Once this file has been sorted, the index is then printed: first on lined paper for proofing, and then on tape for preparing the camera-ready copy.

As the index is being formatted for the quarterly issue, the annual author subset file is being formed. The format of the annual records is:

1. Author name (132 bytes)
2. Publication year (2 bytes)
3. Clearinghouse accession number (6 bytes)

Total record length is 140 bytes.

As in the other indexes, at the end of the year, the quarterly subset files are combined to form the annual index.

Conversion of Document Number Index

The conversion of document number index is formed from the punched cards prepared by the resume print program. The cards punched by the resume program contain the Clearinghouse accession number, ED number, and resume section page number. The information necessary for this index is the Clearinghouse accession number and ED number. Since these decks are rather small, the deck for each index can be saved and the information from one quarter to the next can be combined into one annual index.

The index itself consists of two parts: the conversion of Clearinghouse accession numbers to ED or MP numbers, and the conversion of ED or MP numbers to Clearinghouse accession numbers. Both of these sections may be cumulative.

The annual conversion of document number index may be formed by combining the Clearinghouse accession numbers to ED numbers sections of the indexes from each quarterly issue. The annual index contains only the Clearinghouse to ED number conversion.

Edit Programs

Since errors may be made in preparing the data base, two points have been identified as requiring an edit capability. The first is the MASTER tape--if this tape is to serve the key role it was designed for, then it must be free of errors. The second point is the annual subset files. With the possibility of an error being missed until the annual is to be produced, an edit capability should be available, avoiding the need to rerun a quarter's publication for the sake of an error in the subset file. The MASTER tape is eventually corrected; however, the second edit allows publication schedules to proceed.

Master Edit Program

Some resumes will be found to contain errors during proofing of the printout. The first step in editing is to read the list of Clearinghouse accession numbers on the tape to identify the documents needing correction. The remaining documents are copied onto a new tape. Incorrect resumes are corrected by punching new data cards to replace those in error. The resumes of corrected documents are added to the end of the new tape. The MASTER edit program also allows the deletion of documents in order to obtain the desired paging in the resume section.

Annual Subset Edit Program for Author, Subject, and Supporting Services Indexes

One program will handle the editing for the author, subject, and supporting services indexes. There are three phases to this program: (1) replace any records which have errors in them, (2) delete any excess records, and (3) add any records which have been skipped. For each of the phases, the keyword, publication date, and Clearinghouse accession number are read. The keyword is either the subject, the author, or the code for the supporting service. In the case of the replacement phase, both the data for the bad document and the data for the correct document are needed.

Annual Subset Edit Program for Document Number Index

A separate program is needed to edit the document number index. The program is similar to the edit program for the other subset files; however, due to the change in content and number of fields, a separate program is needed.

Use of QUERY Program

QUERY, primarily designed for retrieving information from the tapes in the ERIC format, is also useful for retrieving information from Clearinghouse collections provided that they are also in ERIC format.

The first use of QUERY in this software package is for retrieving information from RIE for the Clearinghouse publications. Many documents that the Clearinghouse may want to use will already have appeared in RIE. Once these records have been extracted, the ERIC format is converted to the Clearinghouse bibliographic format and abstract format by use of a program listed in Appendix E and flowcharted in Appendix F. The bibliographic tape then can be used with the index programs to prepare the index sections of the publication. The bibliographic tape and abstract tape are then used with the resume print program to prepare the resume section for publication.

QUERY can also be used for obtaining documents pertaining to a particular topic. These documents could then be processed through the programs, and publications containing resume sections and indexes could be prepared. When the Clearinghouse has published its own documents, the question arises: How do they search these documents? When the document base becomes large enough that computer applications are feasible, the software package contains programs to convert the document base to the ERIC format (Appendix E and F). With the document in the ERIC format, the tape can be searched with QUERY, just as the original ERIC tapes could be searched.

III

SUMMARY AND RECOMMENDATIONS

Summary of the Project

The purpose of this project was to develop a software package for the production of Clearinghouse abstract journal publications and creation of magnetic tape files for use in document retrieval. The programs in the package are written in PL/I for use on IBM 360 or other computers with a PL/I compiler. Also needed are keypunch machines, three tape drives, card reader, and line printer. A SORT/MERGE program is needed to supplement the routines prepared especially for this software package.

Programs in the software package facilitate preparation of camera-ready copy of resume sections, and subject, author, supporting service, document number, and document number conversion indexes. Search tapes may also be created for use in computer searching of the Clearinghouse collection or repackaging of the document base in other forms.

The procedures require that document data be stored on magnetic tapes. Input to the bibliographic and abstract tapes is by punched data cards for non-RIE documents and through format conversion of RIE documents identified in a QUERY search of the ERIC tapes. Data on bibliographic and abstract tapes are merged into a resume section which is formatted and printed. The bibliographic tape is used to format and print subject, author, supporting services, and document number indexes. Cards created as an outcome of the resume section print program are used to format and print the document number conversion index. A search tape is created by merging bibliographic and abstract tapes.

The utility of the complete software package has been proven in production of the Fall 1970 issue of Abstracts of Research and Related Materials in Vocational and Technical Education (ARM). Portions of the package have been used in production of Spring, Summer, and Fall (1970) issues of Abstracts of Instructional Materials in Vocational and Technical Education (AIM); the Spring and Summer (1970) issues of ARM; and the 1969-70 Annual Index.

The software package has been documented and is suitable for installation in other Clearinghouses. Feasibility has been checked with five other Clearinghouses (i.e., Counseling and Personnel Services, Educational Media and Technology, Reading, Rural Education and Small Schools, Science and Mathematics Education).

Recommendations

It is recommended that ERIC Clearinghouses be advised of availability of the software package. This announcement should encourage installation by any Clearinghouse which has a document collection sufficiently large to warrant computer retrieval capacity and by Clearinghouses which produce bibliographies and/or abstract journals from RIE and Clearinghouse document collections. This will require duplication of software documentation appended to this report.

It is also recommended that provisions be made for training sessions, visitation, or consultation necessary to successful installation and utilization of the software package.

It is further recommended that additional procedures be developed so that the bibliographic tape be created when a document is first accessioned by the Clearinghouse. Additional routines could be developed which would permit duplicate checking, preparation of the Acquisition Data List, source checking, and production of control file cards.

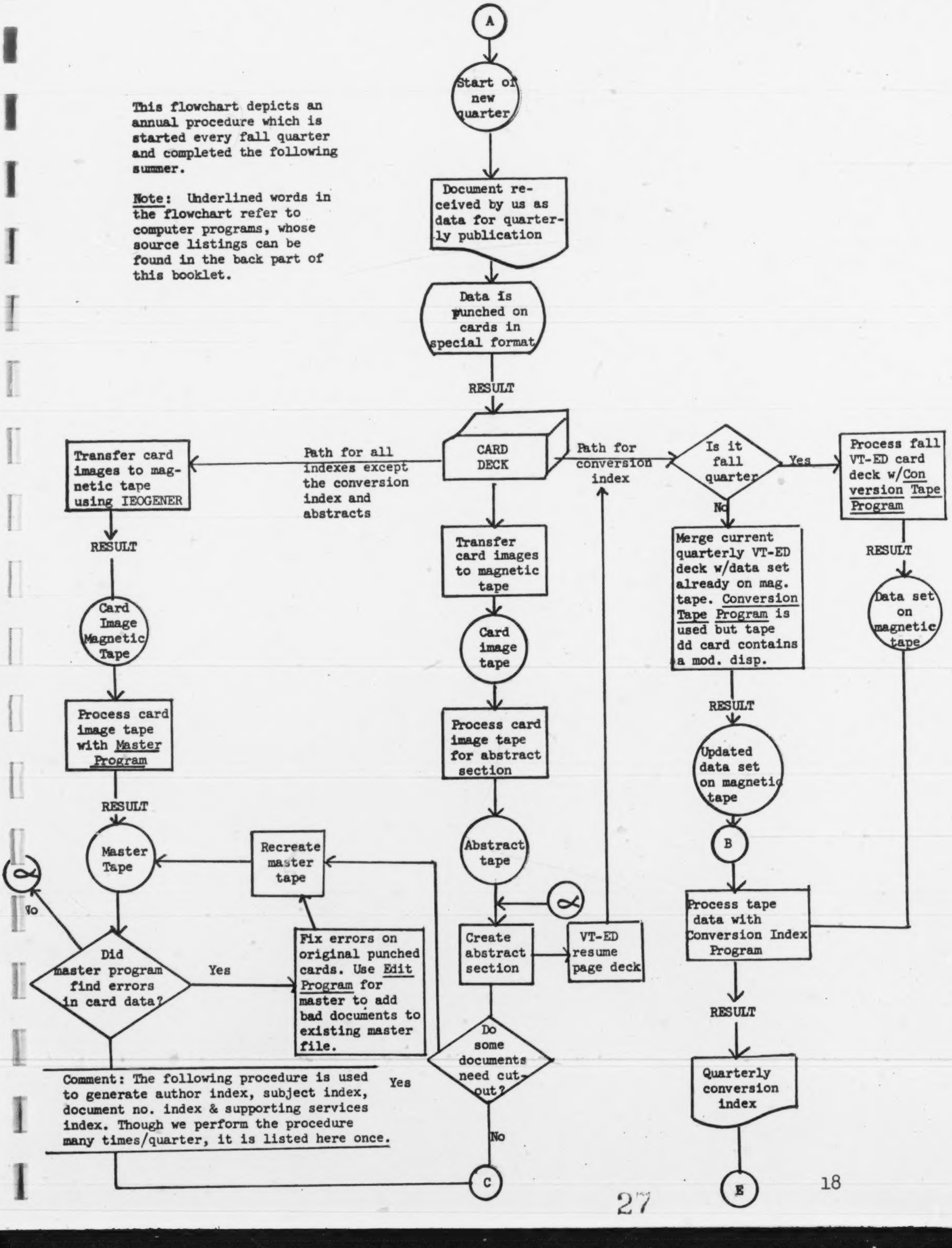
Flowchart of the System

(The System Is the Same for AIM and ARM)

Figure 2

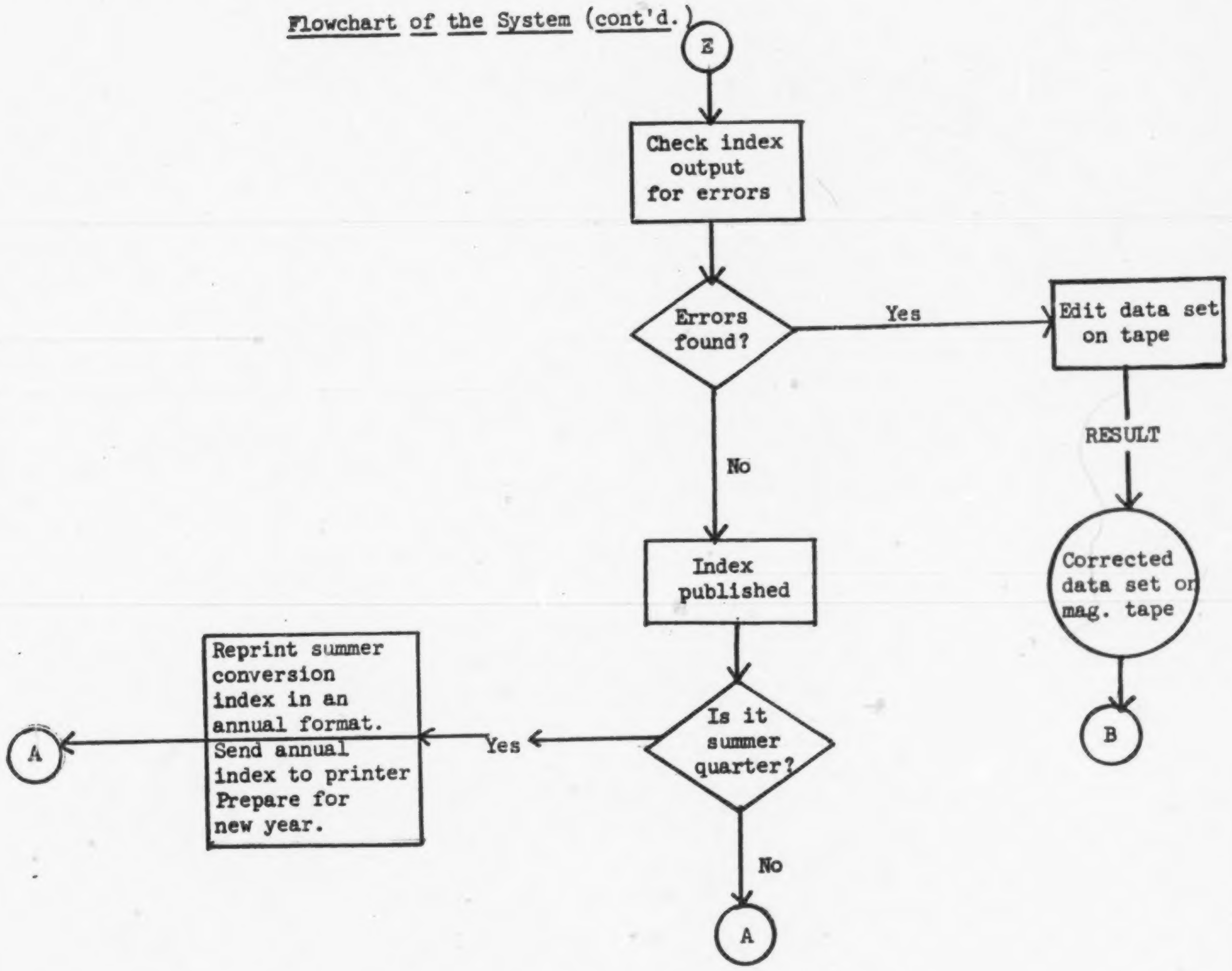
This flowchart depicts an annual procedure which is started every fall quarter and completed the following summer.

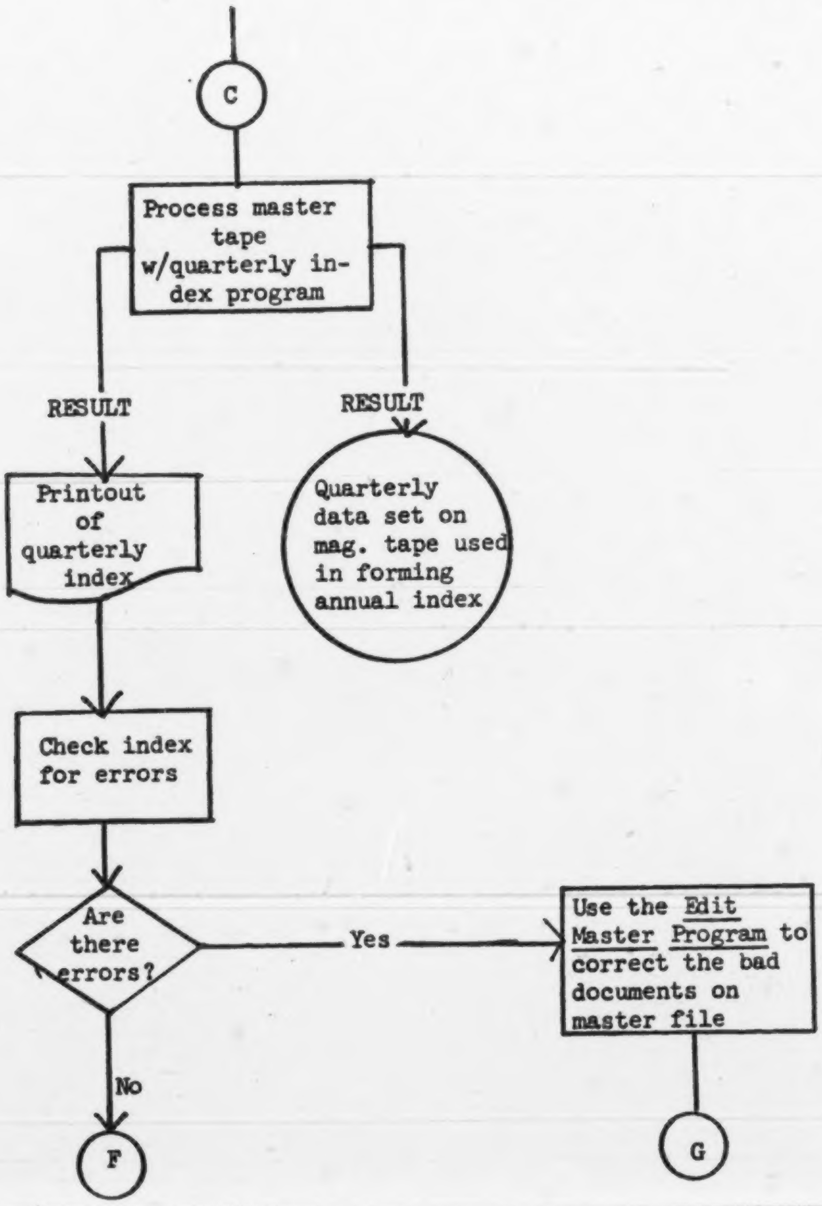
Note: Underlined words in the flowchart refer to computer programs, whose source listings can be found in the back part of this booklet.



A
Start of new quarter
Document received by us as data for quarterly publication
Data is punched on cards in special format
RESULT
CARD DECK
Path for all indexes except the conversion index and abstracts
Transfer card images to magnetic tape using IEOGENER
RESULT
Card Image Magnetic Tape
Process card image tape with Master Program
RESULT
Master Tape
Did master program find errors in card data?
No
α
Yes
Fix errors on original punched cards. Use Edit Program for master to add bad documents to existing master file.
Recreate master tape
Comment: The following procedure is used to generate author index, subject index, document no. index & supporting services index. Though we perform the procedure many times/quarter, it is listed here once.
Transfer card images to magnetic tape
Card image tape
Process card image tape for abstract section
Abstract tape
α
Create abstract section
VT-ED resume page deck
Do some documents need cut-out?
Yes
No
C
Path for conversion index
Is it fall quarter
Yes
No
Process fall VT-ED card deck w/Conversion Tape Program
RESULT
Data set on magnetic tape
Merge current quarterly VT-ED deck w/data set already on mag. tape. Conversion Tape Program is used but tape dd card contains a mod. disp.
RESULT
Updated data set on magnetic tape
B
Process tape data with Conversion Index Program
RESULT
Quarterly conversion index
E

Flowchart of the System (cont'd.)


E
Check index output for errors
Errors found?
Yes
Edit data set on tape
RESULT
Corrected data set on mag. tape
B
No
Index published
Is it summer quarter?
Yes
Reprint summer conversion index in an annual format. Send annual index to printer Prepare for new year.
A
No
A


C
Process master tape w/quarterly index program
RESULT
RESULT
Printout of quarterly index
Quarterly data set on mag. tape used in forming annual index
Check index for errors
Are there errors?
Yes
Use the Edit Master Program to correct the bad documents on master file
No
F
G

Flowchart of the System (cont'd.)

Figure 2

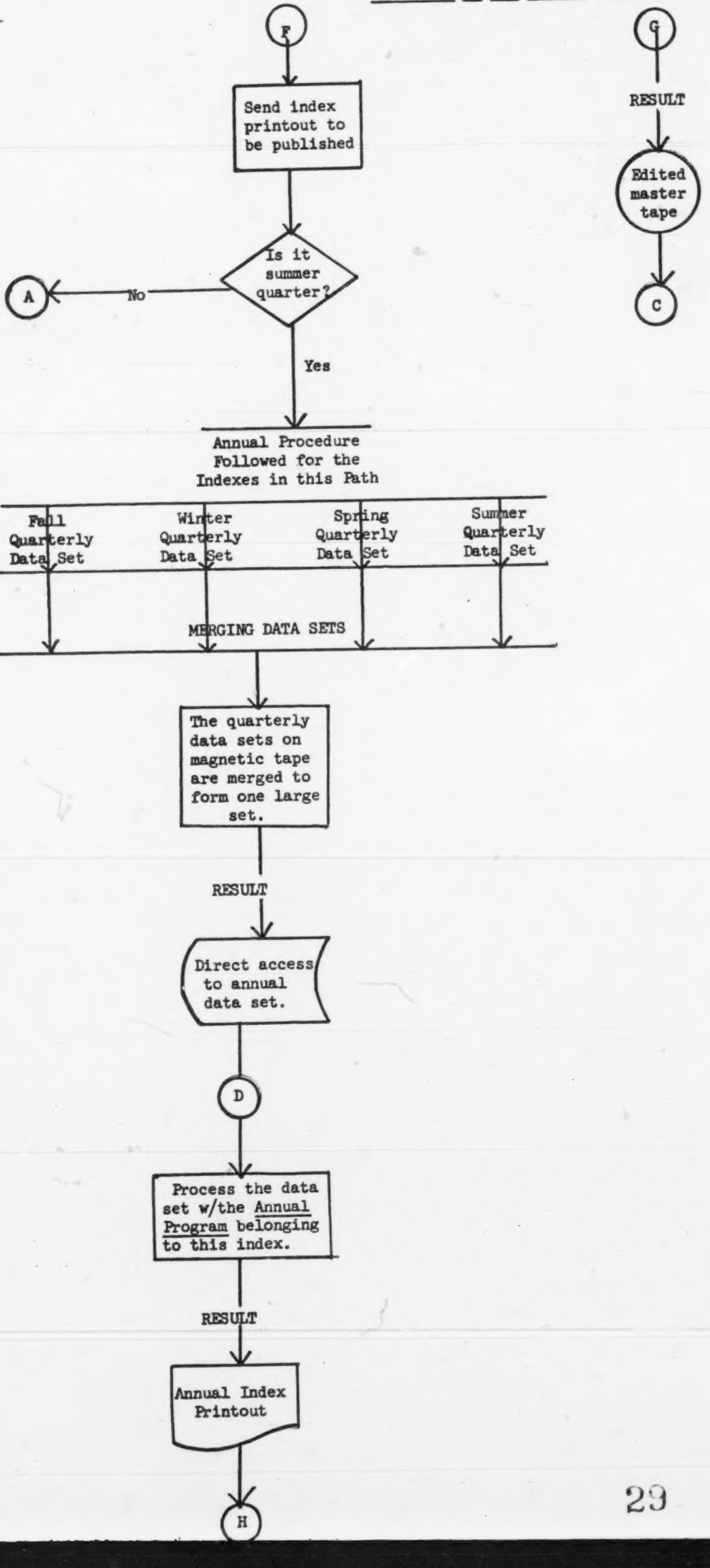
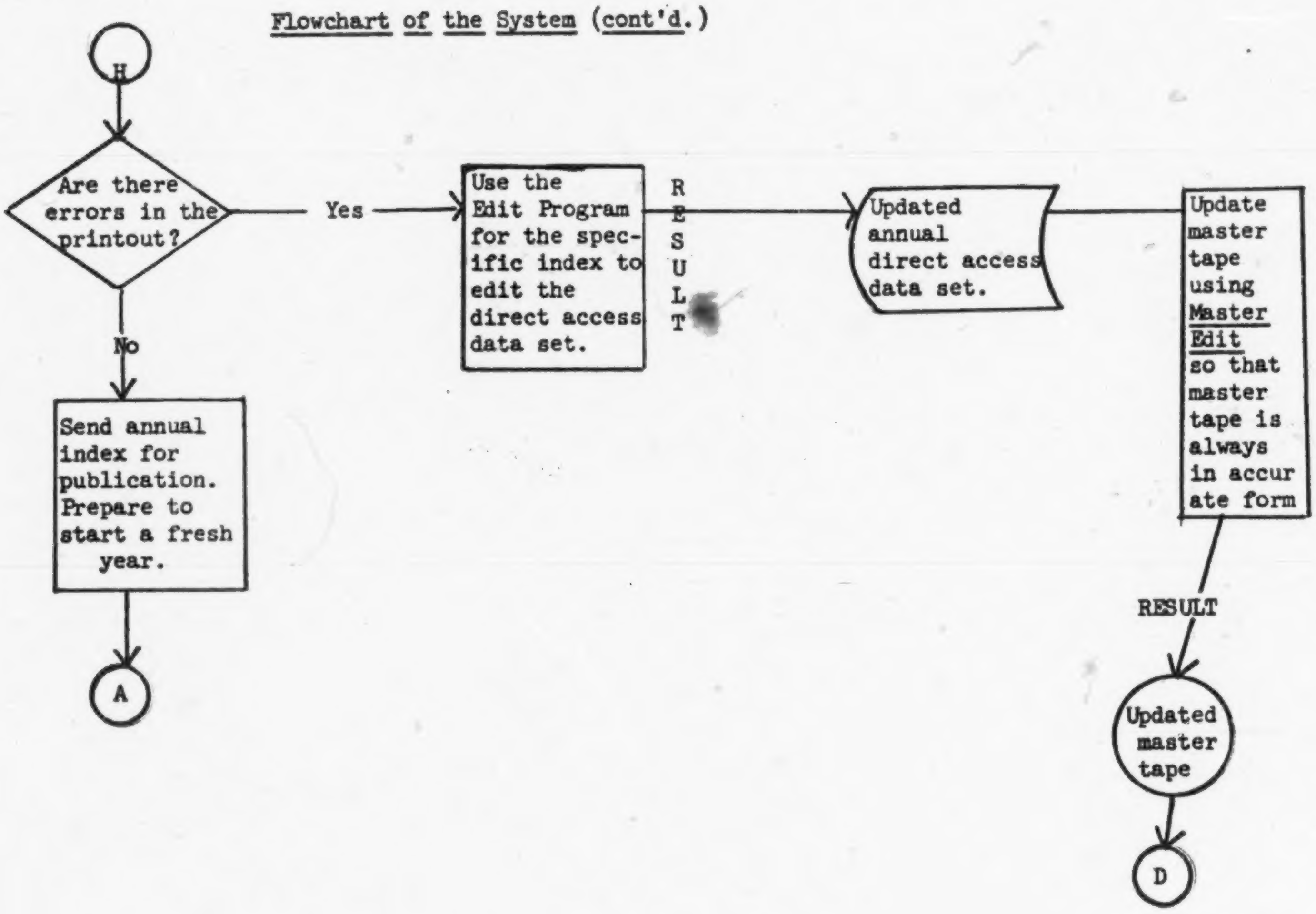

F
Send index printout to be published
Is it summer quarter?
No
A
Yes
Annual Procedure Followed for the Indexes in this Path
Fall Quarterly Data Set
Winter Quarterly Data Set
Spring Quarterly Data Set
Summer Quarterly Data Set
MERGING DATA SETS
The quarterly data sets on magnetic tape are merged to form one large set.
RESULT
Direct access to annual data set.
D
Process the data set w/the Annual Program belonging to this index.
RESULT
Annual Index Printout
H
G
RESULT
Edited master tape
C

Figure 2

Flowchart of the System (cont'd.)


H
Are there errors in the printout?
Yes
Use the Edit Program for the specific index to edit the direct access data set.
RESULT
Updated annual direct access data set.
Update master tape using Master Edit so that master tape is always in accurate form
RESULT
Updated master tape
D
No
Send annual index for publication. Prepare to start a fresh year.
A

APPENDICES

Number		Page
A.	KEYPUNCHING PROCESS	2
B.	JOB CONTROL LANGUAGE	7
C.	CONTROL CARDS	36
D.	ERROR MESSAGES	53
E.	PROGRAM LISTINGS	58
F.	PROGRAM AND SYSTEM FLOWCHARTS	210
G.	PAGE SAMPLES	264

APPENDIX A

KEYPUNCHING PROCESS

Appendix A

The Process of Keypunching for Clearinghouse Documents

The Punching Process

The VT number uses the first 10 columns of the punch card, followed by the line and card number on the next 4 columns, with the data starting on column 15.

Example: VT 000 0000011Data.........

The bibliographic cards have a zero added before the line number and a 1 or 2 after the line number that designates the card number. This gives the card a 4 number digit. For example: line 20 goes to 0201 - 0202, etc., depending on how many cards are needed for the title. If there are more than two cards needed, the number goes to 0211 - 0212, 0221 - 0222, keeping them in the 20 class but building up the series. The 100 and 200 classes have only the card number added, to make the fourth digit. For example: 2001 - 2002.

A program card is used to control the keypunch and automatically reproduces the VT number from card to card. At the end of each number class of information, an end-of-line mark is used to show the end. This is the horizontal bar on the keypunch. Example: If class number 30 went onto only one card it would be

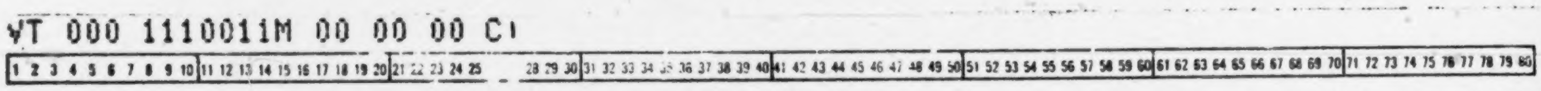
VT 000 0000301Data..........|

or if on two cards, would be

VT 000 0000301Data..........
VT 000 0000302..............|

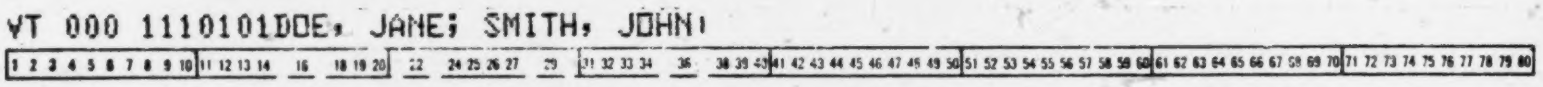
Bibliographic Section

00
01 These two lines are combined into the VT code card. Two of these cards are needed, one for the bib section and one for the abstract section, as these cards are kept separate in the files. The end-of-line mark is also used at the end of this card. Example:



```
VT 000 1110011M 00 00 00 C|
```

10 This card contains the Personal Author data and uses the end-of-line mark. If more than one author appears, they are separated with a semi-colon. Example:



```
VT 000 1110101DOE, JANE; SMITH, JOHN|
```

20 This card contains the Title of Document and also has special editing marks used by the computer. These are short title marks showing the beginning and end of the title wanted for the Subject Index so they are placed at various places within the title cards, and the end-of-line mark is used. The title marks are the "less than" mark at the beginning and the "more than" mark at the end. Example:

```
VT 000 1110201<WALLPAPER AND PAINT>.. WORKBOOK.|
```

30
31
32 These cards are the Corporate Author data cards and uses the end-of-line
33 mark. These names must have a period before the end-of-line mark. Example:
34

```
VT 000 1110301OHIO STATE UNIV., COLUMBUS. DISTRIBUTIVE EDUCATION SERVICE.|
```

```
VT 000 1110311OHIO OHIO STATE UNIV., COLUMBUS. DISTRIBUTIVE EDUCATION MATERIALS
```

```
VT 000 1110312LAB.|
```

40
41 These cards are the Sponsoring Agency data cards and uses the end-of-line
42 mark. Example:

```
VT 000 1110401OHIO STATE DEPT. OF EDUCATION, COLUMBUS|
```

50 The card contains the price line, contract number, grant number, and PA number. There are also special editing marks used in this class and the end-of-line mark. The special edit marks are the tab or ¬ on the punch. These are placed after the price, after the contract number, after the grant number and end-of-line after the PA number. For example: Price ¬ Contract ¬ Grant ¬ PA /. If there are no contract, grant or PA, these 3 marks must still be shown. Example:

```
VT 000 1110501EDRS PRICE MF-$0.25  HC-$1.50¬¬OE12345¬08|
```

```
VT 000 1110501EDRS PRICE MF-$0.25  HC-$1.50¬¬¬|
```

60 This card contains the Report number and the Bureau number. There is also a tab used if one or the other is not available, and the end-of-line mark. For example: 0601Report¬| or 0601¬Bureau| or if both are present the tab is used to separate them. Example:

```
VT 000 1110601PUB-123¬|
```

70 This card contains the availability of the document and the end-of-line mark. Example:

```
VT 000 1110701THE CENTER FOR VOCATIONAL AND TECHNICAL EDUCATION, THE OHIO STATE
VT 000 1110702UNIVERSITY, 1900 KENNY ROAD, COLUMBUS, OHIO 43210 ($2.00).|
```

80 The Journal Citation is on this card. There is also an end-of-line mark. Example:

```
VT 000 1110801AMERICAN VOCATIONAL JOURNAL; V49 N10 P49-51 DECEMBER 1970|
```

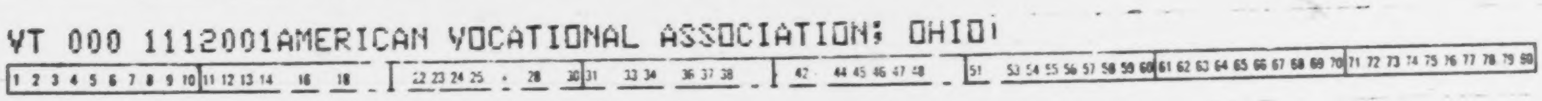
90 The date, pages and any descriptive note are on this line and editing marks are also used. The first seven spaces starting with column 15 are used for the date, a tab to separate from the pages, the page number and another tab to separate from the descriptive note and the end-of-line mark. Example: 0901Date¬Page¬| or 0901Date¬Page¬Descriptive note|. The 2 tabs must always appear. Example:

```
VT 000 111090131NOV69¬112P.¬PAPER PRESENTED AT AMERICAN VOCATIONAL ASSOCIATION C
VT 000 1110902ONFERENCE|
```

100 This class contains the Descriptors and uses the end-of-line mark. The descriptors are separated with a semi-colon and the major descriptors are designated with an asterisk. Example:

```
VT 000 1111001*TEXTBOOKS; *STUDY GUIDES; HIGH SCHOOLS; POST SECONDARY EDUCATION;
VT 000 1111002 *INDIVIDUAL INSTRUCTION|
```

200 This is the Identifier class and uses the end-of-line mark. These are also separated with a semi-colon. Example:



```
VT 000 1112001AMERICAN VOCATIONAL ASSOCIATION; OHIO
```

Abstract Section

The abstract section is handled the same way as the bibliographic section. The cards carry the VT number and line class starts with 300, and carries the 1 or 2 card number. Example: 3001 - 3002, 3011 - 3012, etc. No zero is added before the line number. These cards also have a second run through the keypunch so the zone punch can be added to all lower case letters.

Adding the Lower Case Punch

Once the cards are punched correctly the first time, any line having lower case is run back through the punch a second time adding the zone punches that will cause the letters to be dropped to lower case on the main computer run for proofing and publication. This data is designated in lower case on white-forms. The zone punches are the numerical "zero" zone punch, which punches in the zero row and prints a zero over the letters A through I; the 12 punch which punches in the 12 row and prints an ampersand or "&" over letters J through R; and the 11 punch which punches in the 11 row and prints a dash over letters S through Z. The numeric shift must be held down on these punches.

After a document is finished, and end-of-document card is placed at the end of the bib section and the abstract to designate to the computer that that document is finished. The EOD card has a tab (¬) in column 1, with the rest of the card blank.

APPENDIX B

JOB CONTROL LANGUAGE

	page
RIE TO AIM/ARM	8
MASTER	10
ABSTRACT	11
MASTER SORT	12
RESUME PRINT	13
MASTER EDIT	14
SUBJECT QUARTERLY	15
AUTHOR QUARTERLY	17
SUPPORTING SERVICES QUARTERLY	19
DOCUMENT NUMBER QUARTERLY	21
CONVERSION OF DOCUMENT NUMBER	23
SUBJECT ANNUAL	25
AUTHOR ANNUAL	27
SUPPORTING SERVICES ANNUAL	29
DOCUMENT NUMBER ANNUAL	32
EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICES	33
DOCUMENT NUMBER EDIT	34
RIE FORMAT	35

NOTE: J.C.L. is for either object module or load module. Both forms are illustrated. If need to recompile, check with a programmer.

RIE TO AIM/ARM

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
/*
//VTSIN DD *
/*
//WORK DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(,PASS),DCB=(RECFM=V,
// LRECL=3736,BLKSIZE=3740)
//VTABS DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DCB=(RECFM=V,LRECL=2000,
// BLKSIZE=2004),DISP=(,PASS)
//RIE DD UNIT=2400,VOL=SER=ERIC25,DISP=OLD,DSN=VTERIC,DCB=(RECFM=V,
// LRECL=3617,BLKSIZE=3625)
```

WORK is the data set for the MASTER records. The normal storage media will be magnetic tape.

VTABS is the data set for the abstract records.

RIE is the ddname for the data set containing the original records in RIE format.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//RIETOUT EXEC PGM=VTBIB,TIME=2
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
/*
//VTSIN DD *
/*
```

RIE TO AIM/ARM (cont'd.)

Load Module (cont'd.)

```
//WORK DD UNIT=2400,VOL=SER=ERICB,DISP=(,PASS),
// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740)
//VTABS DD UNIT=2400,VOL=SER=ERICA,DISP=(,PASS),
// DCB=(RECFM=V,LRECL=2000,BLKSIZE=2004)
//RIE DD UNIT=2400,VOL=SER=ERIC25,DISP=OLD,
// DSN=VTERIC,DCB=(RECFM=V,LRECL=3617,BLKSIZE=3625)
```

10.

MASTER

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2

//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR

//SYSLOUT DD SYSOUT=A

//SYSLIN DD *   MASTER obj. follows

          obj. deck

/*

//SYSPRINT DD SYSOUT=A

//ERRE DD SYSOUT=A   Print error messages

//MASTER DD UNIT=2400,VOL=SER=ERIC04,LABEL=(5,SL),DSN=FARM,

// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),DISP=(NEW,KEEP)

//SYSIN DD *  Control cards and VT-ED deck

         see MASTER listing for card information

/*

//CARDS DD *   bibliographic cards

/*
```

The card //CARDS DD * may be replaced with a card referring to a tape if cards are on tape, e.g.:

```
//CARDS DD UNIT=2400,VOL=SER=ERIC18,LABEL=(10,SL),DISP=(OLD,KEEP)

// DSN=CFAIM T,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
```

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,UNIT=2314,VOL=SER=CL0091,DISP=SHR

//FARM EXEC PGM=MASTER,TIME=2

//SYSPRINT DD SYSOUT=A

//ERRE DD SYSOUT=A

//MASTER DD  definition of tape to hold records

//SYSIN DD *    control cards

/*

//CARDS DD   definition of tape holding cards or * if card input

/*
```

ABSTRACT

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *  object deck for abstract
//SYSPRINT DD SYSOUT=A
//SORTIN DD UNIT=2400,VOL=SER=ERIC27,DSN=FLRMABS,DISP=(,PASS),
// DCB=(RECFM=V,LRECL=2000,BLKSIZE=2004)
//SYSIN DD *   control data
/*
//CARDS DD *   abstract data
/*
```

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//ABS EXEC PGM=ABSGEN,TIME=2
//SYSPRINT DD SYSOUT=A
//SORTIN DD UNIT=2400,VOL=SER=ERIC27,DSN=FLRABS,DISP=(,PASS),
// DCB=(RECFM=V,LRECL=2000,BLKSIZE=2004)
//SYSIN DD *   control data
/*
//CARDS DD *   abstract data
/*
```

12.

MASTER AND ABSTRACT SORT

```
//JOBLIB DD DSN=SYS1.JOBLIB,DISP=SHR

//SORT EXEC PGM=IERRC000,TIME=1

//SYSOUT DD SYSOUT=A

//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR

//SORTIN DD UNIT=2400,VOL=SER=ERIC04,DSN=SUAIM,LABEL=(4,SL),DISP=OLD,

//DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740)

//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(,PASS),

//DCB=(REDFM=V,LRECL=3736,FLKSIZE=3740)

//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SYSIN DD *
```

The control cards in the SYSIN data set are of the form

```
    SORT FIELDS=(x,2,CH,A,5,10,CH,A),SIZE=E500
    END
```

The number used in place of x depends on the information being sorted.

Abstract data	use 17
ARM data	use 23
AIM data	use 21

RESUME PRINT

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *
//SYSPRINT DD SYSOUT=A
//OUT DD SYSOUT=A,DCB=(RECFM=VA,LRECL=137,GLKSIZE=141)
//NEWMAS DD UNIT=2400,DISP=NEW,DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),
                                 VOL=SER=ERIC
//MASTER DD DSN=*.SORT.SORTOUT,DISP=(OLD,PASS)
//PUNCH DD DUMMY,DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//SORTOUT DD UNIT=2400,VOL=SER=ERIC21,DSN=SUAMABS,DCB=(REDFM=V,
// LRECL=2000,BLKSIZE=2004),DISP=OLD
//SYSIN DD *
```

NEWMAS is the MASTER data set with the resume pages added.

MASTER is the bibliographic data without page numbers

PUNCH is for the VT-ED resume page decks.

SORTOUT is the abstract data to be read.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//ABSPRT EXEC PGM=APRINT,TIME=2
//SYSPRINT DD SYSOUT=A
//OUT DD UNIT=2400,DISP=NEW,DCB=(RECFM=V,LRECL=3736,VLKSIZE=3740),
                              VOL=SER=ERIC
//MASTER DD DSN=*.SORT.SORTOUT,DISP=(OLD,PASS)
//PUNCH DD DUMMY, DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//SORTOUT DD UNIT=2400,VOL=SER=ERIC21,DSN=SUAMABS,DCB=(RECFM=V,
// LRECL=2000,BLKSIZE=2004),DISP=OLD
//SYSIN DD *
```

MASTER EDIT

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2

//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR

//SYSLOUT DD SYSOUT=A

//SYSLIN DD *   object deck for MASTER edit

/*

//SYSPRINT DD SYSOUT=A

//ERRE DD SYSOUT=A     print error messages

//SYSIN DD *   control cards and VT-ED deck

/*

//CARDS DD *   bibliographic information for edits

/*

//MAST1 DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(4,SL),DSN=WNTAIM,

// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),VOL=SER=ERIC02

//MASTER DD UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL)DSN=WNTAIM,

// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),VOL=SER=ERIC04
```

MAST 1 is the data set to be edited.

MASTER will contain the edited bibliographic file.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR

//MASED EXEC PGM=MASEDT,TIME=2

//SYSPRINT DD SYSOUT=A

//ERRE DD SYSOUT=A     print error messages

//SYSIN DD *   control cards and VT-ED deck

/*

//CARDS DD *   bibliographic information for eidts

/*

//MAST1 DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(4,SL),DSN=WNTAIM,

// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),VOL=SER=ERIC02

//MASTER DD UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL)DSN=WNTAIM,

// DCB=(RECFM=V,LRECL=3736, BLKSIZE=3740),VOL=SER=ERIC04
```

SUBJECT QUARTERLY

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *  quarterly subject object module
/*
//SYSPRINT DD SYSOUT=A
//ERRE DD SYSOUT=A
//MASTER DD UNIT=2400,DSN-WNTAIM,DISP=(OLD,KEEP),LABEL=(1,SL),
// DCB=(RECFM=V,LRECL=3736,FLKSIZE-3740),VOL=SER=ERIC04
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1)),
// DSN=UNSUB,DCB=(RECFM=V,LRECL=573,BLKSIZE=577)
//SORTOUT DD UNIT==SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1)),
//DSN=STSUB,DCB=(RECFM=V,LRECL=573,BLKSIZE=577)
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
//SUB DD UNIT=2400,DSN=AIMF2,DISP=(NEW,KEEP),LABEL=(1,SL),
//SYSIN DD * *** SYSIN DATA ON CARDS ***    control for page format
/*
```

OUTS is the index printout. When camera-ready is prepared, this card is replaced by DD card defining the tape data set.

SUBJECT QUARTERLY (cont'd.)

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR

//SUBJ EXEC PGM=QSUBJ,TIME=4

//SYSPRINT DD SYSOUT=A

//ERRE DD SYSOUT=A

//MASTER DD UNIT=2400,DSN=WNTAIM,DISP=(OLD,KEEP),LABEL=(1,SL),

// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),VOL=SER=ERIC04

//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR

//SORTIN DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1)),

// DSN=UNSUB,DCB=(RECFM=V,LRECL=573,BLKSIZE=577)

//SORTOUT DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1),

//DSN=STSUB,DCB=(RECFM=V,LRECL=573,BLKSIZE=577)

//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02      "

//SORTWK03      "

//SORTWK04      "

//SORTWK05      "

//SORTWK06      "

//SYSOUT DD SYSOUT=A

//OUTS DD SYSOUT=A,DCB=(RECFBM=FA,LRECL=133,BLKSIZE=133)

//SUB DD UNIT=2400,DSN=AIMF2,DISP=(NEW,KEEP),LABEL=(1,SL),

//SYSIN DD * *** SYSIN DATA ON CARDS ***      control for page format

/*
```

AUTHOR QUARTERLY

1. Object Module

```
//QAUTH EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *   object deck for q. author
//SYSPRINT DD SYSOUT=A
//MASTER DD UNIT=2400,VOL=SER=ERIC02,DISP=OLD,DSN=SPRARM
//SORTIN DD UNIT=SYDA,SPACE=(CYL,(1,1)),DCB=(XECFM=V,LRECL=514,BLKSIZE=568)
//SORTOUT DD UNIT=2400,DSN=SPAUT,DCB=*.SORTIN,DISP=(KEEP),
//VOL=SER=ERIC20
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SYSOUT DD SYSOUT=A
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//OUT DD SYSOUT=A,DCB=(RECFM=VA,LRECL=133,BLKSIZE=137)
//SYSIN DD *   format control data
/*
```

OUT is index being created.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//QAUTH EXEC PGM=QAUTHOR,TIME=2
//SYSPRINT DD SYSOUT=A
//MASTER DD UNIT=2400,VOL=SER=ERIC02,DISP=OLD,DSN=SPRARM
//SORTIN DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DCB=(Xecfm=V,LRECL=514,BLKSIZE=568)
//SORTOUT DD UNIT=2400,DSN=SPAUT,DCB=*.SORTIN,DISP=(KEEP),
```

AUTHOR QUARTERLY (CONT'D.)

```
// VOL=SER=ERIC20

//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SYSOUT DD SYSOUT=A

//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR

//OUT DD SYSOUT=A,DCB=(RECFM=VA,LRECL=133,BLKSIZE=137)

//SYSIN DD *   format control data

/*
```

SUPPORTING SERVICES QUARTERLY

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *   supporting services object module
/*
//SYSPRINT DD SYSOUT=A
//ERRE DD SYSOUT=A
//MASTER DD UNIT=2400,DSN=WNTAIM,LABEL=(4,SL),DISP=(OLD,KEEP),
// DCB=(RECFM=V,LRECL=3736,BLKSIZE3740),BOL=SER=ERIC02
//SORTLIB DD DSN=STS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1)),
// DSN=UNSUB,DCB=(RECFM=V,LRECL=444,BLKSIZE=448)
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
//SUB DD UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL),DSN=AIMF2,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=SER=ERIC09
//SYSIN DD *   format control cards
/*
```

OUTS is the index being created

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
```

SUPPORTING SERVICES QUARTERLY (cont'd.)

```
//QSS EXEC PGM=QSUPSER,TIME=2
//SYSPRINT DD SYSOUT=A
//ERRE DD SYSOUT=A
//MASTER DD UNIT=2400,DSN=WNTAIM,LABEL=(4,SL),DISP=(OLD,KEEP),
// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),VOL=SER=ERIC 02
//SORTLIB DD DSN=STS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(1,1)),
// DSN=UNSUB,DCB=(RECFM=V,LRECL=444,BLKSIZE=448)
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
//SUB DD UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL),DSN=AIMF2,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=SER=ERIC09
//SYSIN DD *   format control cards
/*
```

DOCUMENT NUMBER QUARTERLY

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2

//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR

//SYSLOUT DD SYSOUT=A

//SYSLIN DD *   document number object module

/*

//SYSPRINT DD SYSOUT=A

//SORTLIB DD DSNAME=SYS1.SORTLIB,DISP=SHR

//SORTIN DD DSNAME=INSORT,UNIT=SYSDA,DISP=(NEW,DELETE),

//SORTOUT DD DSNAME=OTSORT,UNIT=SYSDA,DISP=(NEW,DELETE),

//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SYSOUT DD SYSOUT=A

//MASTER DD DSN=WNTAIM,UNIT=2400,VOL=SER=ERIC04,DISP=OLD,LABEL=(1,SL)

//ANNUAL DD DSN=ANNFIL,UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL),

//          DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=SER=ERIC13

//INDIX DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=133)

//SYSIN DD *   format control cards

/*
```

INDIX is the INDEX printout.
ANNUAL is annual subset file.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR

//DOCNUM EXEC PGM=DOCNO,TIME=2

//SYSPRINT DD SYSOUT=A
```

DOCUMENT NUMBER QUARTERLY (cont'd.)

```
//SORTLIB DD DSNAME=SYS1.SORTLIB,DISP=SHR

//SORTIN DD DSNAME=INSORT,UNIT=SYSDA,DISP=(NEW,DELETE),

//SORTOUT DD DSNAME=OTSORT,UNIT=SYSDA,DISP=(NEW,DELETE),

//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SYSOUT DD SYSOUT=A

//MASTER DD DSN=WNTAIM,UNIT=2400,VOL=SER=ERIC04,DISP=OLD,

//          LABEL=(1,SL)

//ANNUAL DD DSN=ANNFIL,UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL),

//          DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=SER=ERIC13

//INDIX DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=133)

//SYSIN DD *   format control cards

/*
```

CONVERSION OF DOCUMENT NUMBER

1. Object Deck

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2

//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR

//SYSLOUT DD SYSOUT=A

//SYSLIN DD *   conversion object deck

/*

//SYSPRINT DD SYSOUT=A

//SYSIN DD *   format control deck

/*

//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)

//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR

//SORTIN DD UNIT=2400,DISP=(OLD,KEEP),DSN=CONVRT,LABEL=(2,SLO,

// VOL=SER=ERIC04,DCB=(RECFM=FB,LRECL=20,BLKSIZE=200)

//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE),

// DSN=TEMPST,DCB=(RECFM=FB,LRECL=20,BLKSIZE=200)

//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)

//SYSOUT DD SYSOUT=A
```

OUTS is the index output.
SORTIN is accumulated VT-ED cards.
This program is used for both the quarterly and the annual conversion of document number index.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
```

CONVERSION OF DOCUMENT NUMBER (cont'd.)

```
//CONDOC EXEC PGM=CONVER,TIME=1
//SYSPRINT DD SYSOUT=A
//SYSIN DD *  format control deck
/*
//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=(OLD,KEEP),DSN=CONVRT,LABEL=(s,SLO,
// VOL=SER=ERIC04,DCB=(RECFM=FB,LRECL=20,BLKSIZE=200)
//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE),
// DSN=TEMPST,DCB=(RECFM=FB,LRECL=20,BLKSIZE=200)
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SYSOUT DD SYSOUT=A
```

SUBJECT ANNUAL

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *    annual subject object deck
/*
//SYSPRINT DD SYSOUT=A
//SYSIN DD *     format control cards
/*
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(1,SL),DSN=SB1,
//SORTOUT DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(4,2)),
// DSN=SORTED,DCB=(RECFM=FB,LRECL=140,BLKSIZE=1400)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL(3))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL(3))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL(3))
//SORTWK04 DD UNIT=SYSDA,SPACE=(CYL(3))
//SORTWK05 DD UNIT=SYSDA,SPACE=(CYL(3))
//SORTWK06 DD UNIT=SYSDA,SPACE=(CYL(3))
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=665)
```

OUTS is the data set for the index.

SUBJECT ANNUAL (cont'd.)

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//SUBJ EXEC PGM=ASUBJ,TIME=3
//SYSPRINT DD SYSOUT=A
//SYSIN DD *     format control cards
/*
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(1,SL),DSN=SB1,
//SORTOUT DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(4,2)),
// DSN=SORTED, DCB=(RECFM=FB,LRECL=140,BLKSIZE=1400)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK04 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK05 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK06 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=665)
```

AUTHOR ANNUAL

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *
//SYSPRINT DD SYSOUT=A
//STEPLIB DD DSN=SYS1.JOBLIB,DISP=SHR
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTOUT DD UNIT=SYSDA, SPACE=(CYL,(1,1)),
//DCB=(RECFM=V,LRECL=564,BLKSIZE=568)
//SORTIN DD UNIT=2400,DCB=*.SORTOUT,VOL=(PRIVATE,RETAIN,SER=ERIC17),
// DISP=(OLD,PASS),DSN=ASRT
// DD UNIT=2400,VOL=SER=ERIC17,DCB=*.SORTOUT,DISP=(OLD,PASS),LABEL=2,
// DSN=AIMSPAUT
// DD UNIT=2400,VOL=SER=ERIC20,DSN=ARMSPAUT,DCB=*.SORTOUT,DISP=OLD
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//OUT DD SYSOUT=A
//SYSIN DD *
/*
```

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
```

AUTHOR ANNUAL (cont'd.)

```
//AAUTH EXEC PGM=AAUTHOR,TIME=2
//SYSPRINT DD SYSOUT=A
//STEPLIB DD DSN=SYS1.JOBLIB,DISP=SHR
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DCB=(RECFM=V,LRECL=564,BLKSIZE=568)
//SORTIN DD UNIT=2400,DCB=*.SORTOUT,VOL=(PRIVATE,RETAIN,SER=ERIC17),
// DISP=(OLD,PASS),DSN=ASRT
// DD UNIT=2400,VOL=SER=ERIC17,DCB=*.SORTOUT,DISP=(OLD,PASS).LABEL=2
// DSN=AIMSPAUT
// DD UNIT=2400,VOL=SER=ERIC20,DSN=ARMSPAUT,DCB=*.SORTOUT,DISP=OLD
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSOUT DD SYSOUT=A
//OUT DD SYSOUT=A
//SYSIN DD *
/*
```

SUPPORTING SERVICES ANNUAL

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *    annual s.s. object deck
/*
//SYSPRINT DD SYSOUT=A
//SYSIN DD *     format control cards
/*
//SORTLIB DD DSN==SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(1,SL),DSN=TESTSB,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),
//SORTOUT DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(4,2)),
// DSN=SORTED,DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK04 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK05 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SORTWK06 DD UNIT=SYSDA,SPACE=(CYL,(3))
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
```

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//ASSR EXEC PGM=ASYPSRV,TIME=3
//SYSPRINT DD SYSOUT=A
```

SUPPORTING SERVICES ANNUAL (cont'd.)

```
//SYSIN DD *     format control cards
/*
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(1,SL),DSN=TESTSB,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),
//SORTOUT DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(4,2)),
// DSN=SORTED,DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(3)),
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(3)),
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(3)),
//SORTWK04 DD UNIT=SYSDA,SPACE=(CYL,(3)),
//SORTWK05 DD UNIT=SYSDA,SPACE=(CYL,(3)),
//SORTWK06 DD UNIT=SYSDA,SPACE=(CYL,(3)),
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A,DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
```

DOCUMENT NUMBER ANNUAL

1. Object Module

```
//VTISSUE EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSIN DD *     VT-issue object deck
/*
//SYSPRINT DD SYSOUT=A
//SYSIN DD *     control parameters
/*
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=OLD,DSN=ANSUB1,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=SER=ERIC
//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DCB=*.SORTIN
//SORTWK01 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK04 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK05 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SORTWK06 DD UNIT=SYSDA,SPACE=(TRK,20,,CONTIG)
//SYSOUT DD SYSOUT=A
//OUTS DD SYSOUT=A
```

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//VTISSUE EXEC PGM=ADNO,TIME=1
//SYSPRINT DD SYSOUT=A
//SYSIN DD *     control parameters
/*
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN DD UNIT=2400,DISP=OLD,DSN=ANSUB1,
```

DOCUMENT NUMBER ANNUAL (cont'd.)

```
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=SER=ERIC

//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DCB=*.SORTIN

//SORTWK01 DD UNIT SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK02 DD UNIT SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK03 DD UNIT SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK04 DD UNIT SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK05 DD UNIT SYSDA,SPACE=(TRK,20,,CONTIG)

//SORTWK06 DD UNIT SYSDA,SPACE=(TRK,20,,CONTIG)

//SYSOUT DD SYSOUT=A

//OUTS DD SYSOUT=A
```

EDIT OF VOCATIONAL, AND SUPPORTING SERVICES, SUBJECT, AND AUTHOR

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *    edit object deck
/*
//FIX DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(1,SL),DSN=QRTVC1,
// VOL=(PRIVATE,RETAIN,SER=ERIC03)
//TEMP DD UNIT=2400,DISP (NEW,KEEP),LABEL=(1,SL),DSN=QRTVC1,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=(,RETAIN,SER=ERIC20)
```

FIX is the data set to be edited.

TEMP is the edited data set.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//EDIT EXEC PGM=SUBJSSA,TIME=2
//SYSPRINT DD SYSOUT=A
//SYSIN DD *   control and edit data
/*
//FIX DD UNIT=2400,DISP=(OLD,KEEP),LABEL=(1,SL),DSN=QRTVC1,
// VOL=(PRIVATE,RETAIN,SER=ERIC03)
//TEMP DD UNIT=2400,DISP=(NEW,KEEP),LABEL=(1,SL)DSN=QRTVC1,
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800),VOL=(,RETAIN,SER=ERIC20)
```

DOCUMENT NUMBER EDIT

1. OBJECT MODULE

```
// EXEC PGM=LOADER,PARM='LET;TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLOUT DD SYSOUT=A
//SYSLIN DD *    edit object deck
/*
//DISC DD UNIT=SYSDA,DSN=&&EDTOUT,DISP=(OLD,PASS),
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800)
//TAPE DD UNIT=2400,VOL=SER=ERIC30,DISP=(NEW,PASS),
// DSN=EDITAPE,DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800)
//CARDS DD *    data
/*
```

DISC is the data set to be edited.

TAPE is the edited data set.

2. LOAD MODULE.

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//EDIT EXEC PGM=VTISSUE,TIME=2
//SYSPRINT DD SYSOUT=A
//CARDS DD *    edit data
/*
//DISC DD UNIT=SYSDA,DSN=&&EDTOUT,DISP=(OLD,PASS),
// DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800)
//TAPE DD UNIT=2400,VOL=SER=ERIC30,DISP=(NEW,PASS),
// DSN=EDITAPE,DCB=(RECFM=FB,LRECL=18,BLKSIZE=1800)
```

RIE FORMAT

1. Object Module

```
//FARM EXEC PGM=LOADER,PARM='LET',TIME=2
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=SHR
//SYSLIN DD *
/*
//SYSPRINT DD SYSOUT=A
//MASTER DD UNIT=2400,VOL=SER=ERIC26,DISP=(MOD,KEEP),DSN=ARIM6970,
// DCB=(RECFM=V,LRECL=6996,BLKSIZE=7000)
//RIE DD UNIT=2400,VOL=SER=ERIC04,DSN=WNTAIM,
// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),DISP=(OLD,PASS)
//ABSTRACT DD UNIT=2400,VOL=SER=ERIC,DISP=OLD,
// DCB=(RECFM=V,LRECL=2000,BLKSIZE=2004)
```

MASTER is the final data set. The format will be the same as RIE and this will be the search data set.

RIE is the data set containing bibliographic data.

ABSTRACT is the abstract records for the appropriate records.

2. Load Module

```
//JOBLIB DD DSN=ERICLIB,DISP=SHR
//RIEFORM EXEC PGM=VTTORIE,TIME=1
//SYSPRINT DD SYSOUT=A
//MASTER DD UNIT=2400,VOL=SER=ERIC26,DISP=(MOD,KEEP),DSN=ARIM6970
// DCB=(RECFM=V,LRECL=6996,BLKSIZE=7000)
//RIE DD UNIT=2400,VOL=SER=ERIC04,DSN=WNTAIM,
// DCB=(RECFM=V,LRECL=3736,BLKSIZE=3740),DISP=OLD
//ABSTRACT DD UNIT=2400,VOL=SER=ERIC,DISP=OLD
// DCB=(RECFM=V,LRECL=2000,BLKSIZE=2004)
```

APPENDIX C

CONTROL DATA CARDS

page

RIE TO AIM/ARM 37

MASTER 38

ABSTRACT 39

RESUME PRINT 40

MASTER EDIT 41

SUBJECT QUARTERLY 42

AUTHOR QUARTERLY 43

SUPPORTING SERVICES QUARTERLY 44

DOCUMENT NUMBER QUARTERLY 45

CONVERSION OF DOCUMENT NUMBER 46

SUBJECT ANNUAL 47

AUTHOR ANNUAL 48

SUPPORTING SERVICES ANNUAL 49

DOCUMENT NUMBER ANNUAL 50

EDIT OF SUBJECT, AUTHOR, & SUPPORTING SERVICES 51

DOCUMENT NUMBER EDIT 52

RIE TO AIM/ARM

There are two card data sets for this program.

1. SYSIN

The data is input in a free format. The first character is a blank or an 'A'. The blank will use columns 23 and 24 of the code card for the abstract code. A letter 'A' will use columns 20 and 21 for the abstract code.

Next is the numeric giving the maximum number of documents to be extracted from the tape. The third item is the 1 to 7 character string ased to denot the issue to be published.

2. VTSIN

This data set consists of an accession number in columns 1 through 10 in the form VT 999 999, and the form, field and category in columns 15 through 32. This code will be in the bibliographic record and the two characters designated by the SYSIN card will be in the abstract record.

MASTER

A. CARD 1: all fields are F(3,0)

These are highest line values for the various sections:

Variable		Section	Current Value (max. names)
1.	NA	Number of Personal Authors	2
2.	N10	Personal author	10
3.	NT	Title	22
4.	NC	Corporate author	34
5.	NS	Sponsoring agency	42
6.	N50	Price section	50
7.	N60	Report and bureau	60
8.	N70	Availability	71
9.	N80	Journal citation	80
10.	N90	Date, page, desc. note	91
11.	N100	Descriptor	104
12.	N200	Identifier	202

B. CARD 2: NUMBER OF DOCUMENTS TO BE PROCESSED (USED AS A LIMIT NOT ACTUAL COUNT) right justify in col. 1-5

C. CARD 3:

1. col. 1, A for AIM, blank for ARM

2. col. 2-3, SEASON 'F ', 'W ', 'SP', or 'SU'

3. col. 4-7, current year 19__.

D. CARDS 4-?

1. col. 1-10, VT number (i.e. Clearinghouse accession number)

2. col. 11-30, ED or accession number

3. col. 31-34, resume page

E. Blank card

ABSTRACT TAPE CREATION

Data consists of one card with an A in column 1 if the abstracts should contain the sixth and seventh characters of the code. Column 1 contains a blank if the ninth and tenth characters are to be used.

RESUME SECTION PRINT PROGRAM

The data set is contained in three parts:

Part 1: one card containing A_OR_BLNK='A'; or A_OR_BLNK='b̸'

('A' for AIM, 'b̸' for ARM)

Part 2: several cards in the form variable=value used for print control

Variable	Usage	Current Value
PRICE_COL	Print column for price data	9
CONTRACT_COL	Space from price to CONTRACT number	5
GRANT_COL	Space from contract to grant number	5
PA_COL	Space from contract or grant number to program area	5
VT_ED_SKIP	Space between VT and ED number	5
TOTAL_COLUMNS	Maximum print positions	124
TOTAL_LINES	Length of page	77
PAGE-WIDTH	Actual count of print characters	88
TOP	Indicate right or left page	'RITE'
VT_COL	Print column for VT no.	9
FIRST_LINE	First line of resume section print on first section page	1
CON_1ST_LINE	First line of resume section print on succeeding pages	1
INDENT	Columsn to be left for margin	8
SPACE_LINES	Number of lines to be skipped between title and corporate author, between the descr. note and descriptors, and between the identifiers and abstract.	1
HEADING	Publication	'ARM/FALL 70'
INDEX_PAGE	First page of resume section	2

Part 3: documents to be deleted.

This data is in free format. The first value is number of documents to be deleted. The maximum number is 20. The remainder of this data consist of the clearinghouse accession numbers contained in quotes.

MASTER EDIT

A. Card 1: number of documents (N) to be edited, right justified in columns 1-4.

B. Cards 2-N+1: the clearinghouse accession numbers of documents to be edited in columns 1-10.

C. Card N+2: all fields are F(3,0).

These are the highest line values for the various sections or maximum number of names:

Variable	Section	Current Value
NA	Personal author	2
N10	"	10
NT	Title	22
NC	Corporate author	34
NS	Sponsoring agency	42
N50	Price section	50
N60	Report and Bureau	60
N70	Availability	71
N80	Journal citation	80
N90	Date, page, desc note	91
N100	Descriptors	104
N200	Identifiers	202

D. Card N+3: Total number of documents expected to be processed. This number is an upper limit and the current value is 600.

E. Card N+4: Card to indicate the issue:

1. Column 1 contains an A if AIM and a blank if B
2. Columns 2-3 -- F for FALL, W for WINTER, SP for SPRING, or SU for SUMMER.
3. Columns 4-7 -- the current year 19??

F. Cards N+5 to end of deck: clearinghouse accession number, ED number, and page number of the resume.

SUBJECT QUARTERLY

Data is one continuous stream and varies in length.

Field	Format	Contents	Current Value
1	F(4,0)	First page of index	552
2	F(4,0)	Size of page (total lines)	77
3	F(4,0)	Column width	37
4	F(4,0)	Left hand page margin	9
5	F(4,0)	Right hand page margin	99
6	F(4,0)	Title page line for page number and heading	2
7	F(4,0)	Title page line for word "page"	9
8	F(4,0)	Line of succeeding pages for page number and heading	2
9	F(4,0)	Line of succeeding pages for title	4
10	F(4,0)	Line of succeeding pages for word "page"	7
11	F(4,0)	Spaces between VT and ED numbers	5
12	F(4,0)	Number of columsn in index printout	2
13	F(4,0) repeated no. of cols. times	Margin for each column measured from column one	9,54
14	F(4,0)	Length of title	28
15	A(field 14's value)	Index title	SUBJECT AND IDENTIFIER INDEX
16	F(4,0)	Length of heading	13
17	A(field 16's value)	Heading (e.g. Fall/ARM 70)	WINTER/ARM 69
18	A(1)	Which side is first page on (L for left, R for right)	L

AUTHOR QUARTERLY

Data is in format: variable name = value.

Variable Name	Variable Usage	Current Values
1. #_COLS	Number of columns	2
2. MARGIN	Total print space for authors' name	37
3. LEN_PAGE	Number of print lines	77
4. HEADING	(e.g. AIM/FALL 69)	AIM/WINTER 69
5. TOP	Which side is the page (e.g. TOP='LEFT' the index page number is on left)	LEFT
6. COL SPACE	Character string on blanks to be placed between columns	'b̸b̸b̸b̸b̸'
7. VT_ED_SKIP	Number of spaces to be skipped between VT and ED numbers	5
8. INDEX PAGE	Initial page number	892
9. INITIAL LINE	Line for 'PAGE' on title page	11
10. NEXT INITIAL	Line for 'PAGE' on succeeding pages	9

SUPPORTING SERVICES QUARTERLY

Data is one continuous stream and varies in length.

Field	Format	Contents	Current Values
1	F(4,0)	Initial page	917
2	F(4,0)	Length of pages	77
3	F(4,0)	Column width index printout	37
4	F(4,0)	Left hand margin	9
5	F(4,0)	Right hand margin	93
6	F(4,0)	Title page line number for page number and heading	2
7	F(4,0)	Title page line number for 'PAGE'	9
8	F(4,0)	Succeeding pages line number for page number and heading	2
9	F(4,0)	Succeeding pages line number for title	4
10	F(4,0)	Succeeding pages line number for 'PAGE'	7
11	F(4,0)	Number of spaces between VT and ED numbers	5
12	F(4,0)	Number of columns per page	2
13	F(4,0) repeated no. of cols. times	Column margin from column one	9,54
14	F(4,0)	Length of title	40
15	A(value of field 14)	Index title	VOCATIONAL AND SUPPORTING SERVICES INDEX
16	F(4,0)	Length of heading	13
17	A(value of field 16)	Heading	ARM/WINTER 69
18	A(1)	Indicates whether page number is in left (L) or right (R) margin	R

DOCUMENT NUMBER QUARTERLY

There are two print control cards necessary.

A. CARD 1

Columns	Contents	Current Values
1-3	AIM or ARM	AIM
4-6	line number of first line INDX, SEASON, Year, PAGE	2
7-10	first page number	927
11-13	length or page (i.e. print lines)	77
14-16	first page left or right (l=left, r=right)	2
17-19	width of margin 1	8
20-22	width of margin 2 (0 if no margin)	5
23-25	width of margin 3 (0 if no margin)	0
26-28	width of column (from VT in VT to E in PAGE)	40
29-31	number of columns	2
32-34	number of spaces between VT number and ED number	5
35-37	line number of line containing "PAGE"	7
38-40	line number of first VT ED line	9
41-43	line number of title page for "PAGE"	9
44-46	line number of title page for VT ED	11

B. CARD 2

Columns	Contents
1-3	length of title
4-6	line number for title
7-80	title

CONVERSION OF DOCUMENT NUMBER

Data is one continuous stream and varies in length. This program is used for both annual and quarterly index.

Field	Format	Contents	Current Values
1	F(4,0)	initial page	600
2	F(4,0)	number of print lines	93
3	F(4,0)	column width	25
4	F(4,0)	left hand margin	9
5	F(4,0)	right hand margin	102
6	F(4,0)	line for page number on title page	2
7	F(4,0)	line for subtitle on title page	9
8	F(4,0)	line on succeeding pages for page number	2
9	F(4,0)	line on succeeding pages for title	4
10	F(4,0)	line on succeeding pages for subtitle	7
11	F(4,0)	lines skipped at bottom of page before footnote	3
12	F(4,0)	number of columns	3
13	F(4,0) repeated no. of cols. times	column margins measured from column one	3,9
14	F(4,0)	number of lines of print per block	42
15	F(4,0)	number of blocks of print per columns on title page	75
16	F(4,0)	number of blocks of print per column on succeeding pages	5
17	F(4,0)	length of title	35
18	A(value of field 17)	index title	CONVERSION OF DOCUMENT NUMBER INDEX
19	F(4,0)	length of heading	13
20	A(value of field 19)	heading stating issue, reason, and year	AIM/ARM 69-70
21	A(1)	Q for quarterly index, A for annual index	A
22	A(1)	L if title page is on left, R if title page is on right	L

SUBJECT ANNUAL

Data is one continuous stream and varies in length.

Field	Format	Contents	Current Value
1	F(4,0)	First page of index	1
2	F(4,0)	Size of page (total lines)	95
3	F(4,0)	Column width	34
4	F(4,0)	Number significant digits for VT number	5
5	F(4,0)	Left hand page margin	9
6	F(4,0)	Right hand page margin	125
7	F(4,0)	Title page line for page number and heading	2
8	F(4,0)	Line on title page directly above first line of index data	10
9	F(4,0)	Line of succeeding pages for page number and heading	2
10	F(4,0)	Line of succeeding pages for title	4
11	F(4,0)	Line on succeeding pages directly above first line of index.	7
12	F(4,0)	Number of columns in index printout	3
13	F(4,0) repeated no. of cols. times	Margin for each column measured from column one	9
14	F(4,0)	Length of title	49
15	A(field 14's value)	Index title	SUBJECT AND IDENTIFIER INDEX
16	A(13)	Heading "AIM/ARM 69-70"	
17	A(1)	L if title page is on left, R if title page is on right	R

AUTHOR ANNUAL

Data is in format: variable name=value.

Variable Name	Variable Usage	Current Value
MARGIN	Width of print	37
VT_WIDTH	Columns allocated to VT no.	5
DATE	Last 2 digits of present year	70
INDEX_PAGE	Initial page number	280
#_COLS	Number of columns	2
PRINT_LINE	Number of print positions across page	95
PAGE_LINES	Length of page	77
HEADING	E.g., "AIM/ARM 69-70"	"AIM/ARM 69-70"
TOP	LEFT if initial page is on left RITE if initial page is on right	"RITE"
FIRST_INITIAL	First line of index on title page	9
NEXT_INITIAL	First line of index on succeeding pages	7
COL_SPACE	String of blanks to be placed between columns	'b̸b̸b̸b̸b̸'

SUPPORTING SERVICES ANNUAL

Data is one continuous stream and varies in length.

Field	Format	Contents	Current Value
1	F(4,0)	First page of index	93
2	F(4,0)	Size of page (total lines)	97
3	F(4,0)	Column width	34
4	F(4,0)	Number significant digits for VT number	5
5	F(4,0)	Left hand page margin	9
6	F(4,0)	Right hand page margin	125
7	F(4,0)	Title page line for page number and heading	2
8	F(4,0)	Line on title page directly above first line of index data	10
9	F(4,0)	Line of succeeding pages for page number and heading	2
10	F(4,0)	Line of succeeding pages for title	4
11	F(4,0)	Line on succeeding pages directly above first line of index	7
12	F(4,0)	Number of columsn in index printout	3
13	F(4,0) repeated no. of cols. times	Margin for each column measured from column one	9
14	F(4,0)	Lenth of title	49
15	A(field 14's value)	Index title	VOCATIONAL AND SUPPORTING SERVICES INDEX
16	A(13)	Heading "AIM/ARM 69-70"	
17	A(1)	L if title page is on left, R if title page is one right	R

DOCUMENT NUMBER ANNUAL

Field	Format	Contents	Current Value
1	F(4)	Number of index page	97
2	F(4)	Maximum length of page	97
3	F(4)	No. of significant digits in VT no.	5
4	F(4)	Left hand margin	10
5	F(4)	Right hand margin	124
6	F(4)	Line for heading and page number	2
7	F(4)	Line for subtitle	11
8	F(4)	Line for heading and page number on succeeding pages	2
9	F(4)	Line for index title on all but title page	4
10	F(4)	Line for subtitle on succeeding pages	8
11	F(4)	Number of columns between VT number and issue in printout	5
12	F(4)	Number of columns between issue and page number in printout	5
13	F(4)	Number of columns in index printout	3
14	Field 13's Value*F(4)	Array of column margins measured from column one.	15, 55, 95
15	F(4)	Number of lines of output in one block	5
16	F(4)	Number of blocks of output on title page	14
17	F(4)	Number of blocks of output on other pages	14
18	F(4)	Length of index title	39
19	A (Field 18's value)	Title of index	*
20	A(13)	Heading	AIM/ARM 67-70
21	A(1)	Whether title page shall be on left or right	R

* VT number, issue, and page number index.

EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICES

Card 1: free format first field a numeric indicating the length of the keyword. (e.g. Author edit the value is 132). This field is followed by a blank and then 4 characters in quotes to indicate the index to be edited. If the index is Subject, use "SUBJ."

Card 2: number of records to be replaced followed by the number of records to be deleted.

Card 3:to end of replacements: keyword, date and VT number of error followed by keyword, date, and VT number of correct record. One set for each document.

Cards following replacement deck: deletion deck containing keyword, date, and VT number for documents to be deleted.

Cards following deletion deck: first a card containing number of documents to be added. This card is followed by a deck of cards containing keyword, date, and VT number for each document to be addded.

DOCUMENT NUMBER EDIT

Format of input to edtindex

VT number	Columns 1-6
Page No. (right justified)	Columns 7-10
Issue	Columns 11-15
Card Code	Column 80

If card code is a blank, all corresponding entries on the input data set will be deleted when written to output data set.

If card code is an A, columns 1-18 of that input card will be used as the entry to be written to the output data set.

Sort Input Cards

Numeric Sort	Columns 1-6

APPENDIX D

ERROR MESSAGES FROM PROGRAMS

page

MASTER 54

ABSTRACT 54

RESUME PRINT 54

MASTER EDIT 55

SUBJECT QUARTERLY 55

AUTHOR QUARTERLY 56

SUPPORTING SERVICES QUARTERLY 56

SUBJECT ANNUAL 56

AUTHOR ANNUAL 56

SUPPORTING SERVICES ANNUAL 57

MASTER PROGRAM MESSAGES

1. CODE NO, IS OUT OF BOUNDS VT ___ ___. Columns 6 and 7 of code are not between 40 and 52. The limits are set in card 3440 of the program.
2. SHORT TITLE MARKS ARE NOT CORRECT IN DOCUMENT VT ___ ___.
3. MISMATCHED CARD NUMBER IN DOCUMENT VT ___ ___.
4. MISSING END OF LINE OR TAB CHARACTER IN VT ___ ___.
5. LENGTH OF VARIABLE OR DIMENSION IS OFF IN VT ___ ___. This message is used for character strings which are too long for present sizes or there are too many entries, e.g. more than 5 corporate authors would cause this under current limits.
6. DOCUMENT NO. DOES NOT MATCH ON ONE CARD IN VT ___ ___.
7. EXPECTED INFORMATION MISSING FROM DOCUMENT VT ___ ___. This may be caused by some cards being out of order.
8. NO MATCH MADE ON ED NUMBER FOR VT ___ ___. The clearinghouse accession number card for this document is missing.

ABSTRACT

Cards from abstract record indicate there was a mistake in the abstract card deck.

RESUME PRINT

1. ERROR ON MATCH VT ___ ___ VT ___ ___. Trouble in matching bibliographic record with abstract record.
2. ERRORS IN VTS RESUME BIB VT ___ ___ VT ___ ___. Trouble in matching abstract record with bibliographic record.
3. TITLE ERROR bibliographic record. Title marks are not correct in the bibliographic record printed.

MASTER EDIT

1. CODE NO. IS OUT OF BOUNDS VT ___ ___. Columns 6 and 7 of code are not between 40 and 52. The limits are set in card 3440 of the program.

2. SHORT TITLE MARKS ARE NOT CORRECT IN DOCUMENT VT ___ ___.

3. MISMATCHED CARD NUMBER IN DOCUMENT VT ___ ___.

4. MISSING END OF LINE OR TAB CHARACTER IN VT ___ ___.

5. LENGTH OF VARIABLE OR DIMENSION IS OFF IN VT ___ ___. This message is used for character strings which are too long for present sizes or there are too many entries, e.g. more than 5 corporate authors would cause this under current limits.

6. DOCUMENT NO. DOES NOT MATCH ON ONE CARD IN VT ___ ___.

7. EXPECTED INFORMATION MISSING FROM DOCUMENT VT ___ ___. This may be caused by some cards being out of order.

8. NO MATCH MADE ON ED NUMBER FOR VT ___ ___. The clearinghouse accession number card for this document is missing.

SUBJECT QUARTERLY

1. MISSING DIVIDER IN DOCUMENT VT ___ ___. The divider separating the numeric length fields from the character string is missing in the bibliographic record.

2. KEYWORD TOO LARGE IN VT ___ ___. The present limit on the descriptor and identifier length is set at 200 characters and a keyword in the designated document is too long.

3. TROUBLE IN BREAK WITH VT ___ ___. This error message indicates that there is no blank or - to break up the descriptor or identifier for this document.

AUTHOR QUARTERLY

1. INDEX IS WRONG index number. Separator in bibliographic record is missing.
2. PERSONAL AUTHOR WRONG numeric. Number of personal authors exceeds two.
3. CORPORATE AUTHOR WRONG numeric. Number of corporate authors exceeds five.
4. SPONSORING WRONG numeric. Number of sponsoring agencies exceeds three.
5. TITLE ERROR title VT ___ ___. Error in the title marks.

SUPPORTING SERVICES QUARTERLY

MISSING DIVIDER IN DOCUMENT VT ___ ___. The divider seperating the numeric length fields from the character string is missing in the bibliographic record.

SUBJECT ANNUAL

1. NO END OF SUBJECT NAME IN subject VT ___ ___. Special character used for end of subject name is missing from the record.
2. MORE THAN 15 DATES UNDER service field. Present limit on number of dates for a service field is 15 dates.
3. MORE THAN 75 DOCUMENTS UNDER publication date FOR KEYWORD ____.

AUTHOR ANNUAL

Bibliographic record printed out indicates error in record.

SUPPORTING SERVICES ANNUAL

1. MORE THAN 15 DATES UNDER service field. Present limit on number of dates for a service field is 15 dates.

2. MORE THAN 200 DOCUMENTS UNDER publication date FOR KEYWORD _____.

APPENDIX E

PROGRAM LISTINGS

page

RIE TO AIM/ARM 59

MASTER 69

ABSTRACT 84

RESUME PRINT 87

MASTER EDIT 95

SUBJECT QUARTERLY 106

AUTHOR QUARTERLY 120

SUPPORTING SERVICES QUARTERLY 128

DOCUMENT NUMBER QUARTERLY 141

CONVERSION OF DOCUMENT NUMBER 150

SUBJECT ANNUAL 159

AUTHOR ANNUAL 171

SUPPORTING SERVICES ANNUAL 178

DOCUMENT NUMBER ANNUAL 190

EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICES 197

DOCUMENT NUMBER EDIT 200

RIE FORMAT 202

RIE TO AIM/ARM

```
/*
         G E N E R   P R O G R A M

    PURPOSE:  REFORMAT R I E TAPES INTO FORMATS FOR BIBLIOGRAPHIC
              AND ABSTRACT TAPES


         M A S T E R   R E C O R D   F O R M A T
*/

     1.  VT NUMBER          10 BYTES
     2.  CODE NUMBER        17 BYTES
     3.  ED NUMBER          20 BYTES
     4.  PUB. ISSUE          3 BYTES     A OR BLANK AND 2 BYTH QUARTER
     5.  CURRENT YEAR        4 BYTES
     6.  RESUME PAGE         4 BYTES
     7.  PERSONAL AUTHOR 132 BYTHES MAXIMUM    (MAY BE 2 AUTHORS)
     8.  TITLE           396 BYTES MAXIMUM
     9.  CORP. AUTHORS   660 BYTES MAXIMUM     (MAY BE 5 AUTHORS)
    10.  SPONSOR. AGENCY 396 BYTES MAXIMUM     (MAY BE 3 AGENCIES)
    11.  PRICE            66 BYTES MAXIMUM
    12.  CONTRACT NO.     44 BYTES MAXIMUM
    13.  GRANT NO.        44 BYTES MAXIMUM
    14.  PA NO.            2 BYTES IF IT EXISTS
    15.  REPORT NO.       66 BYTES MAXIMUM
    16.  BUREAU NO.       66 BYTES MAXIMUM
    17.  AVAILABILITY    264 BYTES MAXIMUM
    18.  JOURNAL CITAT.  132 BYTES MAXIMUM
    19.  PUBLICATION DATE  7 BYTES             PUBLICATION OF THE DOCUMENT
    20.  NO. OF DOC. PP.  16 BYTES MAXIMUM
    21.  DESCRIPTOR NOTE 247 BYTES MAXIMUM
    22.  DESCRIPTORS     660 BYTES MAXIMUM
    23.  IDENTIFIERS     396 BYTES MAXIMUM
    24.  SEPARATOR BETWEEN CHARACTER FIELDS AND LENGTH INDICATORS 2 VERTI
       EACH VALUE OCCUPIES 4 BYTES UNLESS NOTED THE FIELD CONTAINS THE LE
       THE INDICATED FIELD
    25. NUMBER OF PERSONAL AUTHORS
    27. TITLE
    28. CORPORATE AUTHOR (NUMBER OF)
    29. CORPORATE AUTHORS  0 TO 5 FIELDS
    30. SPONSORING AGENCIES (NUMBER OF)
    31. SPONSORING AGENCIES 0 TO 3 FIELDS
    32. PRICE
    33. CONTRACT
    34. GRANT NO.
    35. PA NO.
    36. REPORT NO.
    37. BUREAU NO.
    38. AVAILABILITY
    26. EACH P. AUTHOR 0 TO 2 FIELDS
    39.  JOURNAL CITATION
    40.  PUBLICATION DATE
    41.  PAGE COUNT
    42.  DESCRIPTIVE NOTE
    43.  DESCRIPTORS
    44.  IDENTIFIERS

 */
 /*
           C O D E        T A B L E
     SECTION              HEX. CODE         DECIMAL CODE          ORDER
ACCESSION NUMBER            0010                16                  1
```

```
     CLEARINGHOUSE NUM.         0011              17                 2
     PROGRAM AREA               0014              20                 3
     PUBLICATION DATE           0017              23                 4
     TITLE                      001A              26                 5
     PERSONAL AUTHOR            001B              27                 6
     DESCRIPTOR                 0023              35                 7
     IDENTIFIER                 0024              36                 8
     E D R S  PRICE             0025              37                 9
     DESCRIPTIVE NOTE           0026              38                10
     ISSUE                      002B              43                11
     ABSTRACT                   002C              44                12
     REPORT                     002D              45                13
     CONTRACT                   002E              46                14
     GRANT NUMBER               002F              47                15
     BUREAU NUMBER              0030              48                16
     AVAILABILITY               0031              49                17
     JOURNAL CITATION           0032              50                18
     INSTITUTION NAME           0080             128                19
     SPONSORING AGENCY          0084             132                20
    */
    GENER: PROCEDURE OPTIONS(MAIN);
    DCL A_OR_BLNK CHAR(1),
          #_DOC FIXED BINARY(15,0),
        (ICODE, JCODE,KCODE) FIXED BINARY(15,0),
        ISSUE CHAR(7) ;

     /*
               INPUT CONTROL CARD FOR AIM OR ARM RECOGNITION,
               NUMBER OF DOCUMENTS, AND ISSUE
     */

        GET FILE(SYSIN) LIST(A_OR_BLNK,#_DOC,ISSUE);
        IF A_OR_BLNK='A' THEN JCODE=6; ELSE JCODE=9;
    BEGIN;
       CLOSE FILE(SYSIN);
        DCL VTS(#_DOC) CHAR(10);
       DCL CODE(#_DOC) CHAR(17),
        BUF2 CHAR(2000) VAR,
             CARD CHAR(80);
    DCL
        CODE_TABLE(20) FIXED BINARY(15,0),  /* WILL CONTAIN BIT CODES   */
        (I,CODES) FIXED BINARY (31,0),
        LENGTHS FIXED BINARY (15,0),
        TITLE CHAR(400) VAR,
        BUFFER CHAR(6996) VAR, /* TO HOLD INPUT STRING */
        OUT_STRING CHAR(2000) VAR;
      DCL ACC_CODE CHAR(2) INITIAL(' &'),
              BUFFER_LENGTH FIXED BINARY(31,0),
       (J, J2, K, LEN_C_AUTHOR(5), LEN_P_AUTHOR(2), LEN_SA(3))
             FIXED BINARY(15,0),
        CHAR_LENGTH(17) FIXED BINARY(15,0),
        #_P_AUTHORS           FIXED BINARY(15,0),
        TITLE_LEN          FIXED BINARY(15,0),
        #_C_AUTHORS
                           FIXED BINARY(15,0),
        #SA
                           FIXED BINARY(15,0),
        LEN_PRICE
                           FIXED BINARY(15,0),
        LEN_CONTRACT
                           FIXED BINARY(15,0),
        LEN_GRANT
```

```
                      FIXED BINARY(15,0),
       LEN_PA
                      FIXED BINARY(15,0),
       LEN_REPORT
                      FIXED BINARY(15,0),
       LEN_BUREAU
                      FIXED BINARY(15,0),
       LEN_AVAIL
                      FIXED BINARY(15,0),
       LEN_JC
                      FIXED BINARY(15,0),
       LEN_DATE
                      FIXED BINARY(15,0),
       LEN_PAGES
                      FIXED BINARY(15,0),
       LEN_DESC_NOTE
                      FIXED BINARY(15,0),
       LEN_DESC
                      FIXED BINARY(15,0),
       LEN_IDENT
                      FIXED BINARY(15,0),
       (BUREAU, PRICE, REPORT) CHAR(66) VAR,
       (AVAIL, DESC_NOTE, JOURNAL, PERSONAL_AUTHOR) CHAR(132) VAR,
       (CORPORATE_AUTHOR, DESCRIPTOR) CHAR(660) VAR,
       (CONTRACT, GRANT) CHAR(44) VAR,
       (IDENTIFIER, SPONSOR_AGENCY) CHAR(396) VAR,
       DATE CHAR(7),
       (VT, ED) CHAR(10),
       PA CHAR(2) VAR,
       PAGES CHAR(16) VAR,
            NEW_CODE FIXED BINARY(15,0);

     /*
            INITIALIZE CODE TABLE
     */

          CODE_TABLE(1)=16;
          CODE_TABLE(2)=17;
         CODE_TABLE(3)=20;
         CODE_TABLE(4)=23;
         CODE_TABLE(5)=26;
         CODE_TABLE(6)=27;
      DO I=7 TO 10;
         CODE_TABLE(I)=I+28;
      END;
     DO I=11 TO 18;
         CODE_TABLE(I)=32+I;
      END;
         CODE_TABLE(19)=128;
         CODE_TABLE(20)=132;

     /*
            E O F CONTROLS
     */

    ON ENDFILE(RIE) GO TO STOP;
    ON ENDFILE(VTSIN) GO TO INPUT;
            ON TRANSMIT(RIE) GO TO INPUT;
       KCODE=0;

     /*
            READ FILE VTSIN INTO FILE CARD AND GENERATE AN ORDERED
```

```
             SET OF VT NUMBERS WITH A CORRESPONDING SET OF ORDERED
             CODES
    */

LOOP: READ FILE(VTSIN) INTO(CARD);
        KCODE=KCODE+1;
        VTS(KCODE)=SUBSTR(CARD,1,10);
        CODE(KCODE)=SUBSTR(CARD,15,17);
      PUT FILE(SYSPRINT) EDIT(KCODE,VTS(KCODE),CODE(KCODE))(X(5),F(4),X(2
           ),A(10),X(2),A(17));
        GO TO LOOP;

    /*
             READ FILE RIE INTO BUFFER
    */

INPUT: READ FILE(RIE) INTO(BUFFER);

    /*
             INITIALIZE VARIABLES
    */

CHAR_LENGTH=0;
PERSONAL_AUTHOR,TITLE,CORPORATE_AUTHOR,SPONSOR_AGENCY,PRICE='';
CONTRACT,GRANT,REPORT,BUREAU,AVAIL,JOURNAL,DESC_NOTE,DESCRIPTOR='';
IDENTIFIER='';
    DATE=' ';
    PA,PAGES='';
#_P_AUTHORS,TITLE_LEN,#_C_AUTHORS,#SA,LEN_PRICE,LEN_CONTRACT=0;
    LEN_AVAIL,LEN_JC,LEN_PAGES,LEN_DESC_NOTE,LEN_DESC,LEN_IDENT=0;
    LEN_DATE=7;
LEN_GRANT,LEN_PA,LEN_REPORT,LEN_BUREAU=0;
LEN_P_AUTHOR=0;   LEN_C_AUTHOR=0;   LEN_SA=0;

    /*
             SCAN BUFFER FOR ITS LENGTH-BUFFER_LENGTH-, INITIALIZE -I-,
             SCAN BUFFER FOR THE LENGTH-CODE-OF THAT PORTION OF THE
             BUFFER THAT APPEARS BEFORE THE SUBSTRING ACC_CODE
    */

             BUFFER_LENGTH=LENGTH(BUFFER);
              I=1;
           CODES=INDEX(BUFFER,ACC_CODE);

    /*
             DUMP INTO NEW_CODE THE FIELD CODE BIT STRING OF THE
             FIELD FROM THE RIE FILE BEING PROCESSED
    */

                        UNSPEC(NEW_CODE)=UNSPEC(SUBSTR(BUFFER,CODES,2));

    /*
             IF THE FIELD CODE IS OUT OF POSITION, GO TO THE INPUT
             OF THE RIE FILE
    */

                      IF CODES<3 THEN GO TO INPUT;

    /*
             DUMP INTO LENGTHS THE BIT STRING GIVING THE TOTAL LENGTH
             OF THE FIELD BEING PROCESSED
    */
```

```
 NEXT: UNSPEC(LENGTHS)=UNSPEC(SUBSTR(BUFFER,CODES-2,2));

 /*
           IF LENGTHS>BUFFER_LENGTH OR THE FIELD HAS NO DATA, WRITE
           LENGTHS, BUFFER_LENGTH, THEN RETURN TO THE INPUT OF THE
           RIE FILE
 */

    IF LENGTHS>BUFFER_LENGTH|LENGTHS<5 THEN DO;
                PUT FILE(SYSPRINT) EDIT(LENGTHS,BUFFER_LENGTH)(SKIP,
                     2 F(4));
               GO TO INPUT;
                                            END;

 /*
           COPY DATA STRING FROM THE FIELD BEING PROCESSED AND STORE
           IT IN THE PROCESSING LOCATION DENOTED AS OUT_STRING
 */

         OUT_STRING=SUBSTR(BUFFER,CODES+2,LENGTHS-4);

 /*
           IF NEW_CODE=CODE_TABLE(I), THEN ENTER THE PROCESSING BLOCK
           AND STORE CHARACTER LENGTH AND LENGTH INDICATOR
 */

              IF NEW_CODE=CODE_TABLE(I) THEN
 DO;

 /*
           PROCESSING BLOCK:  PURPOSE:  GIVEN A CODE VALUE(I.E. -I-) THE
           BLOCK WILL CONSTRUCT THE APPROPRIATE STRING OF DATA FROM THE
           FIELD BEING PROCESSED.  ALONG WITH THE CONSTRUCTION OF NEW
           DATA STRINGS IN THEIR DESIRED NEW FORM, APPROPRIATE LENGTH
           PARAMETERS ARE SPECIFIED FOR EACH NEW STRING CONSTRUCTED.
           NOTE:  IF I=12, ABSTRACTS ARE BUILT, PRINTED OUT, AND STORED
           IN THE FILE VTABS.
 */

    IF I=1 THEN ED='ED '||SUBSTR(OUT_STRING,3,3)||' '||SUBSTR(
         OUT_STRING,6,3);
           ELSE
IF I=2 THEN DO;
VT=SUBSTR(OUT_STRING,1,2)||' '||SUBSTR(OUT_STRING,3,3)||' '||
   SUBSTR(OUT_STRING,6,3);
   IF SUBSTR(VT,1,2)¬='VT' THEN DO:
      PUT FILE(SYSPRINT) EDIT(BUFFER)(SKIP(2),A);
     GO TO INPUT;
                                  END;
  DO ICODE=1 TO KCODE;
        IF VT=VTS(ICODE) THEN GO TO UPCODE;
  END;
  GO TO INPUT;
           END;
          ELSE
    IF I=3 THEN DO;
               PA=OUT_STRING;
               LEN_PA=LENGTHS-4;
               END;
          ELSE
  IF I=4&LENGTHS>5 THEN SUBSTR(DATE,12-LENGTHS)=OUT_STRING;
```

```
           ELSE
  IF I=5 THEN DO:
 TITLE_LEN=LENGTHS-1:
  IF TITLE_LEN<5|TITLE_LEN>400 THEN DO;
        PUT FILE(ERROR) EDIT (BUFFER)(SKIP,A);
        GO TO INPUT:                        END:
TITLE='<'||OUT_STRING||'>..';
                 END;
            ELSE
   IF I=6 THEN DO;
                 J=INDEX(OUT_STRING,';');
                     IF J¬=0 THEN DO;
                        J=INDEX(OUT_STRING,'AND OTHERS');
         IF J=0 THEN J=INDEX(OUT_STRING,'AND OTHERS');
                    IF J=0 THEN DO;
                           #_P_AUTHORS=2:
     LEN_P_AUTHOR(1)=INDEX(OUT_STRING,';')-1;
   LEN_P_AUTHOR(2)=LENGTHS-6-LEN_P_AUTHOR(1);
   PERSONAL_AUTHOR=SUBSTR(OUT_STRING,1,LEN_P_AUTHOR(1))||SUBSTR(
   OUT_STRING,LEN_P_AUTHOR(1)+3,LEN_P_AUTHOR(2));
                                 END;
                     ELSE DO;
                           J=INDEX(OUT_STRING,';');
                           SUBSTR(OUT_STRING,J,1)=',';
                           #_P_AUTHORS=1:
                           LEN_P_AUTHOR(1)=LENGTHS-4:
                           PERSONAL_AUTHOR=OUT_STRING;
                           END;
         END;
                               ELSE DO:
                                 #_P_AUTHORS=1:
                                 LEN_P_AUTHOR(1)=LENGTHS-4;
            PERSONAL_AUTHOR=OUT_STRING;
                     END;
                  END;
            ELSE
   IF I=7 THEN DO;
              DESCRIPTOR=OUT_STRING;
                  LEN_DESC=LENGTHS-4;
                                         END;
            ELSE
   IF I=8 THEN DO;
              IDENTIFIER=OUT_STRING;
         LEN_IDENT=LENGTHS-4;
                  END;
             ELSE
   IF I=9 THEN DO;
                      PRICE=OUT_STRING;
                  LEN_PRICE=LENGTHS-4;
                  END;
      ELSE
    IF I=10 THEN DO;
         J=INDEX(OUT_STRING,'P.');
          IF J<=0 THEN J=INDEX(OUT_STRING,'P.');
                  IF J<LENGTHS-7 THEN DO;
                    DESC_NOTE=SUBSTR(OUT_STRING,J+4);
                  LEN_DESC_NOTE=LENGTH(DESC_NOTE);
                                 END;
                     LEN_PAGES=J+1:
                                    PAGES=SUBSTR(OUT_STRING,1,J+1);
                  IF J<0 THEN DO;
                         LEN_DESC_NOTE=0;
                      LEN_PAGES=0;
                      END;
                 END;
            ELSE
IF I=11 THEN GO TO UPCODE;
      ELSE
IF I=12 THEN DO;
  BUF2=VT||SUBSTR(CODE(ICODE),JCODE,2)||OUT_STRING;
 PUT FILE(SYSPRINT) EDIT(BUF2)(SKIP(3),A);
WRITE FILE(VTABS) FROM(BUF2);
             END;
            ELSE
    IF I=13 THEN DO;
               REPORT=OUT_STRING;
                 LEN_REPORT=LENGTHS-4;
                 END;
           ELSE
    IF I=14 THEN DO;
             CONTRACT=OUT_STRING;
                 LEN_CONTRACT=LENGTHS-4;
                 END;
      ELSE
    IF I=15 THEN DO;
                GRANT=OUT_STRING;
                 LEN_GRANT=LENGTHS-4;
                 END;
      ELSE
    IF I=16 THEN DO;
                BUREAU=OUT_STRING;
                 LEN_BUREAU=LENGTHS-4;
                 END;
      ELSE
     IF I=17 THEN DO;
                 AVAIL=OUT_STRING;
             LEN_AVAIL=LENGTH(AVAIL);
                  END;
       ELSE
     IF I=18 THEN DO;
                  LEN_JC=LENGTHS-4;
               JOURNAL=OUT_STRING;
                  END;
       ELSE
     IF I=19 THEN DO;
                  K=1;
               CORPORATE_AUTHOR=OUT_STRING;
                  J2=1;
                  JLOOP: J=INDEX(CORPORATE_AUTHOR,';');
                         IF J¬=0 THEN DO;
                           LEN_C_AUTHOR(K)=J-J2;
                        K=K+1;
                           J2=J+1;
                           CORPORATE_AUTHOR=SUBSTR(CORPORATE_AUTHOR,1,J-
                           1)||SUBSTR(CORPORATE_AUTHOR,J+2);
                           GO TO JLOOP;
                                       END;
          ELSE IF J2=1 THEN LEN_C_AUTHOR(1)=LENGTH(OUT_STRING);
                        ELSE LEN_C_AUTHOR(K)=LENGTH(OUT_STRING)-J2;
                         #_C_AUTHORS=K;
                  END;
             ELSE
      IF I=20 THEN DO;
                  K=1;
               SPONSOR_AGENCY=OUT_STRING;
                J2=1;
           JSALOOP: J=INDEX(SPONSOR_AGENCY,';');
                    IF J¬=0 THEN DO:
                      LEN_SA(K)=J-J2;
                      K=K+1;
                      J2=J+1;
                       SPONSOR_AGENCY=SUBSTR(SPONSOR_AGENCY,1,J-1)||
                         SUBSTR(SPONSOR_AGENCY,J+2);
                       GO TO JSALOOP;
                                END;
         ELSE IF J2=1 THEN LEN_SA(1)=LENGTH(OUT_STRING);
                     ELSE LEN_SA(K)=LENGTH(OUT_STRING)-J2;
                    #SA=K;
                END;
END;

 /*
          TO CODES, THE LENGTH OF THE FIELD , BEING PROCESSED, THAT
          APPEARS BEFORE THE SUBSTRING ACC_CODE, ADD THE LENGTH OF
          THE ENTIRE FIELD. THIS IS TO BE USED AS A CHECK FOR TERMINAT
          ING THE PROCESSING OF A COMPLETE RECORD.
 */

UPCODE: CODES=CODES+LENGTHS;

 /*
          IF CODES>BUFFER_LENGTH, AN ENTIRE RECORD HAS BEEN PROCESSED.
          PERFORM OUTPUT AND STORAGE, THEN GO TO INPUT OF THE RIE FILE
 */

 IF CODES>BUFFER_LENGTH|I=20 THEN DO;
          PUT STRING(BUFFER) EDIT(VT,CODE(ICODE),FD,ISSUE,
PERSONAL_AUTHOR,TITLE,CORPORATE_AUTHOR,SPONSOR_AGENCY,PRICE,CONTRACT,
GRANT,PA,REPORT,BUREAU,AVAIL,JOURNAL,DATE,PAGES,DESC_NOTE,DESCRIPTOR,
IDENTIFIER,'||',#_P_AUTHORS,(LEN_P_AUTHOR(J) DO J=1 TO #_P_AUTHORS),
TITLE_LEN,#_C_AUTHORS,(LEN_C_AUTHOR(J) DO J=1 TO #_C_AUTHORS),#SA,
(LEN_SA(J) DO J=1 TO #SA),LEN_PRICE,LEN_CONTRACT,LEN_GRANT,LEN_PA,
LEN_REPORT,LEN_BUREAU,LEN_AVAIL,LEN_JC,LEN_DATE,LEN_PAGES,LEN_DESC_NOTE
,LEN_DESC,LEN_IDENT)(A(10),A(17),A(20),A(7),X(4),18 A,28 F(4));
 PUT FILE(SYSPRINT) EDIT(BUFFER)(SKIP,A);
WRITE FILE(WORK) FROM(BUFFER);
  GO TO INPUT;
                               END;

 /*
          DUMP INTO NEW_CODE THE FIELD CODE -ACC_CODE- BIT STRING
 */

     UNSPEC(NEW_CODE)=UNSPEC(SUBSTR(BUFFER,CODES,2));

 /*
     SCAN CODE_TABLE
          FOR VALUES OF I FROM 2 TO 20 CHECK IF NEW=CODE_TABLE(I).
          IF A MATCH IS FOUND , GO TO STATEMENT NAMED NEXT.
          IF FOR A VALUE OF I, NEW_CODE<CODE_TABLE(I), SET I=I-1
          AND GO TO NEXT.
          OTHERWISE, GO TO INPUT OF THE RIE FILE.
 */

     DO I=I+1 TO 20;
               IF NEW_CODE=CODE_TABLE(I) THEN GO TO NEXT;
```

```
        IF NEW_CODE<CODE_TABLE(I) THEN DO;
             I=I-1;
             GO TO NEXT;
                                      END;
       END;
      GO TO INPUT;
 STOP: CLOSE FILE(WORK), FILE(VTABS),FILE(RIE);
 END GENER;
```

MASTER

```
/**********************************************************************/00000010
/*                      M A S T E R   P R O G R A M                   */00000020
/*                                                                    */00000040
/*                                                                    */00000050
/*   PURPOSE:  THIS PROGRAM GENERATES A MASTER TAPE DATA SET FOR      */00000060
/*             THE QUARTERLY AIM OR ARM DATA.  THE PROGRAM CHECKS     */00000070
/*             THE DATA AS THOROUGHLY AS POSSIBLE FOR ERRORS. IF      */00000080
/*             THE DATA FOR A DOCUMENT IS CORRECT, A RECORD IS FORMED */00000090
/*             FOR THE DOCUMENT ON THE MASTER TAPE.  IF THERE ARE     */00000100
/*             ERRORS FLAGGED, AN ERROR MESSAGE IS PRINTED OUT AND NO */00000110
/*             RECORD GOES ON THE MASTER TAPE.  WHEN ALL THE ERRORS   */00000120
/*             HAVE BEEN FLAGGED, CORRECT THE BAD DATA.  THEN CHANGE  */00000130
/*             THE DISPOSITION ON THE MASTER DD CARD TO  -MOD-  AND   */00000140
/*             THESE DOCUMENTS WILL HAVE THEIR RECORDS ADDED ON TO    */00000150
/*             THE END OF THE DATA SET.                               */00000160
/*                                                                    */00000170
/*   DEFINITION OF VARIABLES:                                         */00000180
/*      1. NA -- MAXIMUM NUMBER OF PERSONAL AUTHORS THAT CAN BE FOUND */00000190
/*              IN THAT SECTION.  (CURRENTLY SET AT 2)                */00000200
/*      2. N10 -- MAXIMUM LINE NUMBER FOUND IN THE PERSONAL AUTHOR    */00000210
/*                SECTION. (CURRENTLY SET AT 10)                      */00000220
/*      3. NT  -- MAXIMUM LINE NUMBER FOUND IN THE TITLE SECTION.     */00000230
/*                (CURRENTLY SET AT 22)                               */00000240
/*      4. NC -- MAXIMUM LINE NUMBER FOUND IN THE CORPORATE AUTHOR    */00000250
/*               SECTION. (CURRENTLY SET AT 34).                      */00000260
/*      5. NS -- MAXIMUM LINE NUMBER FOUND IN THE SPONSORING AGENCY   */00000270
/*               SECTION. (CURRENTLY SET AT 42)                       */00000280
/*      6. N50 -- MAXIMUM LINE NUMBER FOUND IN THE PRICE SECTION.     */00000290
/*               (CURRENTLY SET AT 50)                                */00000300
/*      7. N60 -- MAXIMUM LINE NUMBER FOUND IN THE REPORT OR BUREAU   */00000310
/*                NUMBER SECTION. (CURRENTLY SET AT 60)               */00000320
/*      8. N70 -- MAXIMUM LINE NUMBER FOUND IN THE AVAILABILITY       */00000330
/*                SECTION. (CURRENTLY SET AT 71)                      */00000340
/*      9. N80 -- MAXIMUM LINE NUMBER FOUND IN THE JOURNAL CITATION   */00000350
/*                SECTION. (CURRENTLY SET AT 80)                      */00000360
/*     10. N90 -- MAXIMUM LINE NUMBER FOUND IN THE DATE, PAGE, AND    */00000370
/*                DESCRIPTIVE NOTE SECTION. (CURRENTLY SET AT 91)     */00000380
/*     11. N100 -- MAXIMUM LINE NUMBER FOUND IN THE DESCRIPTOR        */00000390
/*                 SECTION. (CURRENTLY SET AT 104)                    */00000400
/*     12. N200 -- MAXIMUM LINE NUMBER FOUND IN THE IDENTIFIER        */00000410
/*                 SECTION. (CURRENTLY SET AT 202)                    */00000420
/*     13. NDOC -- THE NUMBER OF DOCUMENTS BEING PROCESSED BY THE     */00000430
/*                 MASTER PROGRAM FOR THIS ISSUE OF AIM OR ARM.       */00000440
/*     14. AUTHOR -- THE NAME OF A PERSONAL AUTHOR.                   */00000450
/*     15. AVAILABILITY -- THE AVAILABILITY STRING.                   */00000460
/*     16. BD -- THE CHARACTER FOUND IN COLUMN 1 OF THE CARD THAT     */00000470
/*               SEPARATES THE DOCUMENT DATA ON CARDS.  (¬)           */00000480
/*     17. BT -- CHARACTER DENOTING BEGIN SHORT TITLE. (<)            */00000490
/*     18. BUREAU_NO -- THE BUREAU NUMBER.                            */00000500
/*     19. CARD_NO -- CARD IDENTIFICATION NUMBER FOUND IN COLUMN 14   */00000510
/*                    OF THE DATA CARDS.                              */00000520
/*     20. CAT -- STRING WITH ALL THE DOCUMENT'S DATA USED TO MAKE    */00000530
/*                THE RECORD ON TAPE.                                 */00000540
/*     21. CODE_NO -- CODE NUMBER.                                    */00000550
/*     22. CONTRACT_NO -- CONTRACT NUMBER.                            */00000560
/*     23. CORP_AUTHOR -- CORPORATE AUTHOR.                           */00000570
/*     24. CR -- CHARACTER DENOTING END OF INFORMATION ON DATA CARDS. */00000580
/*              (|)                                                   */00000590
/*     25. DESCRIP -- STRING OF DESCRIPTORS.                          */00000600
/*     26. DESC_NOTE -- DESCRIPTIVE NOTE.                             */00000610
/*     27. E -- ARRAY OF ED OR ACCESSION NUMBERS FOR THE DOCUMENTS    */00000620
/*              BEING PROCESSED.                                      */00000630
```

```
    /*     28. ED_NO -- ACCESSION NUMBER.                                 */00000640
    /*     29. ET -- CHARACTER DENOTING END SHORT TITLE. (>)              */00000650
    /*     30. GRANT_NO -- GRANT NUMBER.                                  */00000660
    /*     31. IDENT -- STRING OF IDENTIFIERS.                            */00000670
    /*     32. IND -- INDICATOR WHICH SAYS WHETHER WE WERE AT THE END     */00000680
    /*                OF A DOCUMENT OR NOT WHEN WE HIT AN UNMATCHING      */00000690
    /*                VT NUMBER. WHEN WE HIT THE BEGIN DOCUMENT CARD,     */00000700
    /*                THERE WILL BE A MISMATCHED VT NUMBER FLAGGED. BUT   */00000710
    /*                THERE IS NO REAL ERROR. -IND- TELLS US THIS.        */00000720
    /*     33. IMP -- HELPS POSITION US AT THE BEGINNING OF A NEW         */00000730
    /*                DOCUMENT WHEN THE PREVIOUS ONE IS FINISHED.         */00000740
    /*     34. JOUR_CIT -- JOURNAL CITATION.                              */00000750
    /*     35. LINE_NO -- LINE NUMBER ON THE DATA CARD.                   */00000760
    /*     36. LONG -- INTERMEDIATE CHARACTER STRING.                     */00000770
    /*     37. L_AUTHOR -- LENGTH OF AN AUTHOR'S NAME.                    */00000780
    /*     38. L_AVAILABILITY -- LENGTH OF AVAILABILITY STRING.           */00000790
    /*     39. L_BUREAU_NO -- LENGTH OF BUREAU NUMBER.                    */00000800
    /*     40. L_CONTRACT_NO -- LENGTH OF CONTRACT NUMBER.                */00000810
    /*     41. L_CORP_AUTHOR -- LENGTH OF A CORPORATE AUTHOR NAME.        */00000820
    /*     42. L_DESCRIP -- LENGTH OF DESCRIPTOR STRING.                  */00000830
    /*     43. L_DESC_NOTE -- LENGTH OF DESCRIPTIVE NOTE.                 */00000840
    /*     44. L_GRANT_NO -- LENGTH OF GRANT NUMBER.                      */00000850
    /*     45. L_IDENT -- LENGTH OF IDENTIFIER STRING.                    */00000860
    /*     46. L_JOUR_CIT -- LENGTH OF JOURNAL CITATION.                  */00000870
    /*     47. L_PAGE -- LENGTH OF PAGE STRING.                           */00000880
    /*     48. L_PA_NO -- LENGTH OF PA NUMBER.                            */00000890
    /*     49. L_PRICE -- LENGTH OF PRICE.                                */00000900
    /*     50. L_PUB_DATE -- LENGTH OF PUBLICATION DATE.                  */00000910
    /*     51. L_REPORT_NO -- LENGTH OF REPORT NUMBER.                    */00000920
    /*     52. L_SPONSOR_AG -- LENGTH OF A SPONSORING AGENCY.             */00000930
    /*     53. L_TITLE -- LENGTH OF TITLE.                                */00000940
    /*     54. MASTER -- NAME OF MASTER TAPE.                             */00000950
    /*     55. NO_AUTHOR -- ACTUAL NUMBER OF PERSONAL AUTHORS FOR A       */00000960
    /*                      DOCUMENT.                                     */00000970
    /*     56. NO_CORP_AUTHOR -- ACTUAL NUMBER OF CORPORATE AUTHORS FOR A */00000980
    /*                           DOCUMENT.                                */00000990
    /*     57. NO_DES -- FLAG TO INDICATE IF THERE ARE ANY IDENTIFIERS    */00001000
    /*                   FOR A DOCUMENT.                                  */00001010
    /*     58. NO_RECS -- COUNTS THE NUMBER OF RECORDS ON THE MASTER TAPE.*/00001020
    /*     59. NO_SPONSOR -- ACTUAL NUMBER OF SPONSORING AGENCIES FOR A   */00001030
    /*                       DOCUMENT.                                    */00001040
    /*     60. PAGE -- NUMBER OF PAGES IN THE REPORT.                     */00001050
    /*     61. PAGE_NO -- PAGE NUMBER IN THE AIM OR ARM PUBLICATION ON    */00001060
    /*                    WHICH THE ABSTRACT OF THE DOCUMENT CAN BE FOUND.*/00001070
    /*     62. PA_NO -- PA NUMBER.                                        */00001080
    /*     63. PG -- ARRAY OF PAGE NUMBERS IN THE PUBLICATION ON WHICH THE*/00001090
    /*                ABSTRACTS CAN BE FOUND.                             */00001100
    /*     64. PRICE -- PRICE STRING.                                     */00001110
    /*     65. PUB_DATE -- PUBLICATION DATE.                              */00001120
    /*     66. REPORT_NO -- REPORT NUMBER.                                */00001130
    /*     67. SEASON -- INDICATES AIM OR ARM AND FOR WHAT SEASON THE     */00001140
    /*                    PUBLICATION IS BEING PRINTED.                   */00001150
    /*     68. SEP -- SEPARATER USED IN THE MASTER RECORD (||).           */00001160
    /*     69. SPONSOR_AG -- NAME OF A SPONSORING AGENCY.                 */00001170
    /*     70. STR -- STRING INTO WHICH THE CARD DATA IS READ.            */00001180
    /*     71. TAB -- INDICATES WHETHER A CARD SHOULD BE READ OR NOT.     */00001190
    /*     72. TB -- TAB SEPARATING DATA WHEN MORE THAN ONE PIECE OF DATA */00001200
    /*                APPEARS ON THE SAME LINE.  (¬)                      */00001210
    /*     73. TITLE -- TITLE OF A DOCUMENT.                              */00001220
    /*     74. V -- ARRAY OF VT OR DOCUMENT NUMBERS BEING PROCESSED.      */00001230
    /*     75. VT -- DOCUMENT NUMBER ON FIRST CARD OF DATA FOR A          */00001240
    /*                PARTICULAR DOCUMENT.                                */00001250
```

```
/*    76. VT_NO -- DOCUMENT NUMBER ON SUCCEEDING DATA CARDS FOR     */
/*                 A DOCUMENT.                                       */
/*    77. YEAR -- CURRENT YEAR.                                      */
/*                                                                   */
/*  INPUT TO PROGRAM:                                                */
/*    -SYSIN-, INPUT ON CARDS:                                       */
/*      CARD 1 -- CONTAINS VALUES FOR THE FOLLOWING VARIABLES, ALL IN*/
/*                F(3,0) FIELDS: NA,N10,NT,NC,NS,N50,N60,N70,N80,N90,*/
/*                N100,N200. THESE VARIABLES ARE DEFINED IN THE      */
/*                PRECEDING SECTION AND THEIR CURRENT VALUES ARE LIST*/
/*                AS WELL. CHANGES IN THESE NUMBERS WILL COME THROUGH*/
/*                THE VOCATIONAL CENTER.                             */
/*      CARD 2 -- CONTAINS THE NUMBER OF DOCUMENTS IN THIS QUARTERLY */
/*                DATA BEING PROCESSED. THE CENTER CAN SUPPLY THE    */
/*                VALUE. RIGHT JUSTIFY IN COL 1-5.                   */
/*      CARD 3 -- COL 1 IS THE ISSUE, 'A' FOR AIM OR ' ' FOR ARM.    */
/*                COL 2-3 IS THE SEASON, 'F ' FOR FALL, 'W ' FOR     */
/*                 WINTER, 'SP' FOR SPRING, 'SU' FOR SUMMER.         */
/*                COL 4-7 IS THE CURRENT YEAR, 19__.                 */
/*      CARDS 4-? -- THESE ARE THE VT, ED, RESUME PAGE CARDS. THE    */
/*                   FORMAT FOR EACH CARD IS:                        */
/*                   COL 1-10 -- VT OR DOCUMENT NUMBER.              */
/*                   COL 11 - 30 -- ED OR ACCESSION NUMBER.          */
/*                   COL 31 - 34 -- RESUME PAGE.                     */
/*                   THE LAST CARD IN THIS DECK MUST BE A BLANK CARD.*/
/*    -CARDS-, INPUT ON TAPE OR CARDS:                               */
/*      THE FIRST RUN WITH A NEW SET OF DATA, -CARDS- WILL BE A DD   */
/*      CARD DEFINING THE CARD IMAGE TAPE. MASTER WILL PROBABLY FLAG */
/*      THE BAD DOCUMENTS. CORRECT THE DATA CARDS IN ERROR FOR THESE */
/*      DOCUMENTS. THEN RERUN THE PROGRAM WITH JUST THE DATA CARDS   */
/*      FOR THE FLAGGED DOCUMENTS FOLLOWING A -//CARDS DD *- CARD.   */
/*      MAKE SURE THE MASTER DD CARD HAS A DISPOSITION OF -MOD-. THE */
/*      BAD DOCUMENTS WILL THEN BE ADDED TO THE MASTER RECORDS       */
/*      ALREADY ON TAPE.                                             */
/*                                                                   */
/*  SPECIAL NOTE:                                                    */
/*    IT WILL BE IMPOSSIBLE TO UNDERSTAND THIS MASTER PROGRAM UNLESS */
/*    YOU ARE FAMILIAR WITH 3 THINGS:                                */
/*    (1) THE FORMAT OF THE MASTER RECORD                            */
/*    (2) THE FORMAT OF THE DATA CARDS                               */
/*    (3) THE NATURE OF THE DATA ON THE CARDS                        */
/*                                                                   */
/*********************************************************************/


BEGIN: PROCEDURE OPTIONS (MAIN);
        DECLARE
          DUMMY
              CHARACTER (1),
          (NA,N10,NT,NC,NS,N50,N60,N70,N80,N90,N100,N200)
              FIXED BINARY (31,0),
          NDOC                FIXED BINARY (31,0),
        ;


 /*  READ IN SOME MAXIMUM BOUNDARIES FOUND ON THE DATA CARDS.  */


        GET FILE (SYSIN) EDIT (NA,N10,NT,NC,NS,N50,N60,N70,N80,N90,
            N100,N200,DUMMY) (12 F(3),X(43),A(1));
        GET FILE (SYSIN) EDIT (NDOC,DUMMY) (F(5),X(74),A(1));

```

```
WORK:   BEGIN;
        DECLARE
          AUTHOR(NA)          CHARACTER (50) VARYING,
          AVAILABILITY        CHARACTER ((N70-69)*132) VARYING,
          BD                  CHARACTER (1) INITIAL ('¬'),
          BEG
              CHARACTER (1),
          BT                  CHARACTER (1) INITIAL ('<'),
          BUREAU_NO           CHARACTER ((N60-59)*132/2) VARYING,
          CARD_NO
              FIXED BINARY (31,0),
          CAT                 CHARACTER (3736) VARYING,
          CODE_NO
              CHARACTER (17),
          CONTRACT_NO         CHARACTER ((N50-49)*132/3) VARYING,
          CORP_AUTHOR(NC-29)  CHARACTER (132) VARYING,
          CR                  CHARACTER (1) INITIAL ('|'),
          DESCRIP             CHARACTER ((N100-99)*132) VARYING,
          DESC_NOTE           CHARACTER ((N90-89)*132-13) VARYING,
          E(NDOC)             CHARACTER (20),
          ED_NO               CHARACTER (20),
          ET                  CHARACTER (1) INITIAL ('>'),
          GRANT_NO            CHARACTER ((N50-49)*132/3) VARYING,
          I
              FIXED BINARY (31,0),
          IDENT               CHARACTER ((N200-199)*132) VARYING,
          IMP                 FIXED BINARY (31,0),
          IND                 FIXED BINARY (31,0),
          JOUR_CIT            CHARACTER ((N80-79)*132) VARYING,
          K
              FIXED BINARY (31,0),
          L
              FIXED BINARY (31,0),
          LINE_NO
              FIXED BINARY (31,0),
          LONG                CHARACTER (660) VARYING,
          L_AUTHOR(NA)        FIXED BINARY (31,0),
          L_AVAILABILITY
              FIXED BINARY (31,0),
          L_BUREAU_NO
              FIXED BINARY (31,0),
          L_CONTRACT_NO
              FIXED BINARY (31,0),
          L_CORP_AUTHOR(NC-29) FIXED BINARY (31,0),
          L_DESCRIP
              FIXED BINARY (31,0),
          L_DESC_NOTE
              FIXED BINARY (31,0),
          L_GRANT_NO
              FIXED BINARY (31,0),
          L_IDENT
              FIXED BINARY (31,0),
          L_JOUR_CIT
              FIXED BINARY (31,0),
          L_PAGE
              FIXED BINARY (31,0),
          L_PA_NO
              FIXED BINARY (31,0),
          L_PRICE
              FIXED BINARY (31,0),
          L_PUB_DATE
```

```
       FIXED BINARY (31,0),
   L_REPORT_NO
       FIXED BINARY (31,0),
   L_SPONSOR_AG(NS-39) FIXED BINARY (31,0),
   L_TITLE
       FIXED BINARY (31,0),
   MASTER
       FILE RECORD OUTPUT SEQUENTIAL UNBUFFERED,
   N
       FIXED BINARY (31,0),
   NO_AUTHOR
       FIXED BINARY (31,0),
   NO_CORP_AUTHOR
       FIXED BINARY (31,0),
   NO_DES
       FIXED BINARY (31,0),
   NO_RECS
       FIXED BINARY (31,0),
   NO_SPONSOR
       FIXED BINARY (31,0),
   PAGE
       CHARACTER (16) VARYING,
   PAGE_NO              CHARACTER (4),
   PA_NO
       CHARACTER (2),
   PG(NDOC)             CHARACTER (4),
   PRICE                CHARACTER ((N50-49)*132/2) VARYING,
   PUB_DATE
       CHARACTER (7),
   REPORT_NO            CHARACTER ((N60-59)*132/2) VARYING,
   SEASON               CHARACTER (3),
   SEP                  CHARACTER (2) INITIAL ('||'),
   SEPERATE
       ENTRY (FIXED BINARY (31,0),FIXED BINARY (31,0),
                FIXED BINARY (31,0),,),
   SPONSOR_AG(NS-39) CHARACTER (132) VARYING,
   STR                  CHARACTER (80),
   TAB
       FIXED BINARY (31,0),
   TB                   CHARACTER (1) INITIAL ('¬'),
   TITLE                CHARACTER ((NT-19)*132) VARYING,
   V(NDOC)              CHARACTER (10),
   VT
       CHARACTER (10),
   VT_NO
       CHARACTER (10),
   YEAR
       CHARACTER (4),
   ;

 OPEN FILE (ERRE) PRINT STREAM OUTPUT LINESIZE(125),
      FILE (CARDS) INPUT RECORD,
      FILE (MASTER);
 ON ENDFILE (CARDS) GO TO STOP;
 ON CONVERSION BEGIN;
   PUT FILE (ERRE) EDIT ('CONVERSION ERROR IN DOCUMENT ',VT)
     (SKIP(2),A(29),A(10));
   GO TO SKIP;
   END;


/*  READ IN REMAINING SYSIN CARDS.  */
```

```
       GET FILE (SYSIN) EDIT (SEASON,YEAR,DUMMY)
          (A(3),A(4),X(72),A(1));
ED_LP: DO I = 1 TO NDOC;
          GET FILE (SYSIN) EDIT (V(I),E(I),PG(I),DUMMY)
             (A(10),A(20),A(4),X(45),A(1));
          IF V(I) = '               ' THEN GO TO ASGN;
          K = I;
       END ED_LP;
ASGN:  NO_RECS = 0;
       READ FILE (CARDS) IGNORE (1);

  /*  READ IN LINE 1 WITH THE CODE NUMBER. CHECK FOR ERRORS INCLUDING A
      TO MAKE SURE THE SUB-CODE IS WITHIN BOUNDS. ANY ERRORS FOUND IN A
      DOCUMENT THROUGHOUT THIS PROGRAM WILL CAUSE AN ERROR MESSAGE TO BE
      PRINTED AND A SKIP TO THE BEGINNING OF THE NEXT DOCUMENT.  */


 START:  READ FILE (CARDS) INTO (STR):
         GET STRING (STR) EDIT (VT_NO,LINE_NO,CARD_NO,CODE_NO)
            (A(10),F(3),F(1),A(17));
         IND = 0;
         IMP = 2;
         TAB = 0;
         VT = VT_NO;
         N = INDEX(CODE_NO,CR);
         IF N = 0 THEN GO TO TEST;
         CODE_NO = SUBSTR(CODE_NO,1,N-1);
 TEST:   IF LINE_NO ¬= 1 THEN GO TO ERR;
         IF CARD_NO ¬= 1 THEN GO TO ERR;
         IF (SUBSTR(CODE_NO,6,2)<40) | (SUBSTR(CODE_NO,6,2)>52)
           THEN DO:
                 PUT FILE (ERRE) EDIT ('CODE NO. IS OUT OF BOUNDS',VT)
                   (SKIP(2),A(25),A(10));
                 GO TO SKIP;
                 END:

    /* CHECK FOR ANY PERSONAL AUTHORS AND STORE THEIR NAMES IN -AUTHOR-*/


          NO_AUTHOR = 0;
          CALL SEPERATE(10,N10,1,TLE_SECT,LONG);
 SRCH:    N = INDEX(LONG,':');
          IF N ¬= 0 THEN DO;
                         NO_AUTHOR = NO_AUTHOR + 1:
                         IF NO_AUTHOR > NA THEN GO TO OFF_LNTH;
                         AUTHOR(NO_AUTHOR) = SUBSTR(LONG,1,N-1);
                         L_AUTHOR(NO_AUTHOR) = N - 1;
                         IF (N-1) > 50 THEN GO TO OFF_LNTH;
                         LONG = SUBSTR(LONG,N+2);
                         GO TO SRCH;
                         END;
          NO_AUTHOR = NO_AUTHOR + 1;
          IF NO_AUTHOR > NA THEN GO TO OFF_LNTH;
          N = INDEX(LONG,CR);
          IF N = 0 THEN GO TO MISSCR;
          AUTHOR(NO_AUTHOR) = SUBSTR(LONG,1,N-1);
          L_AUTHOR(NO_AUTHOR) = N - 1;
          IF (N-1) > 50 THEN GO TO OFF_LNTH;
 /*  PULL THE TITLE OFF OF THE DATA CARDS. CHECK FOR AN EVEN NUMBER OF
     AND END SHORT TITLE MARKS IN THE TITLE.  */


TLE_SECT:
           CALL SEPERATE (20,NT,2,CORP_SECT,LONG);
           N = INDEX(LONG,CR);
           IF N = 0 THEN GO TO MISSCR;
           TITLE = SUBSTR(LONG,1,N-1);
           L_TITLE = N - 1;
           I = 0;
           IDENT = TITLE;
RET:       N = INDEX(IDENT,BT);
           L = INDEX(IDENT,ET);
           IF (N=0) & (L=0) & (I¬=0) THEN GO TO CORP_SECT;
           IF (N=0) | (L=0)
             THEN DO;
                   PUT FILE (ERRE) EDIT
                     ('SHORT TITLE MARKS ARE NOT CORRECT IN DOCUMENT ',VT)
                     (SKIP(2),A(46),A(10));
                   GO TO SKIP;
                   END;
           I = I + 1;
           IDENT = SUBSTR(IDENT,L+2);
           GO TO RET;

 /*  CHECK FOR ANY CORPORATE AUTHORS AND STORE NAMES AND NAME LENGTHS I
     ARE FOUND.  */


CORP_SECT:
           NO_CORP_AUTHOR = 0;
CP_LP:     DO I = 30 TO NC;
           CALL SEPERATE (I,I,3,SPON_SECT,LONG);
           NO_CORP_AUTHOR = NO_CORP_AUTHOR + 1;
           N = INDEX(LONG,CR);
           IF N = 0 THEN GO TO MISSCR;
           CORP_AUTHOR(NO_CORP_AUTHOR) = SUBSTR(LONG,1,N-1);
           L_CORP_AUTHOR(NO_CORP_AUTHOR) = N - 1;
           IF (N-1) > 132 THEN GO TO OFF_LNTH;
           END CP_LP;

 /*  CHECK FOR ANY SPONSORING AGENCIES AND STORE NAMES AND NAME LENGTHS
     ARE FOUND.  */


SPON_SECT:
           NO_SPONSOR = 0;
SP_LP:     DO I = 40 TO NS;
           CALL SEPERATE (I,I,4,PRICE_SECT,LONG);
           NO_SPONSOR=NO_SPONSOR + 1;
           N = INDEX(LONG,CR);
           IF N = 0 THEN GO TO MISSCR;
           SPONSOR_AG(NO_SPONSOR) = SUBSTR(LONG,1,N-1);
           L_SPONSOR_AG(NO_SPONSOR) = N - 1;
           IF (N-1) > 132 THEN GO TO OFF_LNTH;
           END SP_LP;
```

```
  /*  PULL THE PRICE OFF OF LINE 50.  */


PRICE_SECT:
        L_CONTRACT_NO,L_GRANT_NO,L_PA_NO = 0;
        CALL SEPERATE (50,N50,5,REP_BUR_SECT,LONG);
        N = INDEX(LONG,TB);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN GO TO OFF_LNTH;
        PRICE = SUBSTR(LONG,1,N-1);
        L_PRICE = N - 1;
        IF (N-1) > ((N50-49)*132/2) THEN GO TO OFF_LNTH;
        LONG = SUBSTR(LONG,N+1);


  /*  CHECK FOR A CONTRACT NUMBER.  */


        N = INDEX(LONG,TB);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN GO TO MISS1;
        CONTRACT_NO = SUBSTR(LONG,1,N-1);
        L_CONTRACT_NO = N - 1;
        IF (N-1) > ((N50-49)*132/3) THEN GO TO OFF_LNTH;
MISS1:  LONG = SUBSTR(LONG,N+1);


  /*  CHECK FOR A GRANT NUMBER.  */


        N = INDEX(LONG,TB);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN GO TO MISS2;
        GRANT_NO = SUBSTR(LONG,1,N-1);
        L_GRANT_NO = N - 1;
        IF (N-1) > ((N50-49)*132/3) THEN GO TO OFF_LNTH;
MISS2:  LONG = SUBSTR(LONG,N+1);


  /*  CHECK FOR A PA NUMBER.  */


        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN GO TO REP_BUR_SECT;
        PA_NO = SUBSTR(LONG,1,N-1);
        L_PA_NO = N - 1;
        IF (N-1) ¬= 2 THEN GO TO OFF_LNTH;


  /*  CHECK FOR A REPORT NUMBER.  */


REP_BUR_SECT:
        L_REPORT_NO,L_BUREAU_NO = 0;
        CALL SEPERATE(60,N60,6,AVL_SECT,LONG);
        N = INDEX(LONG,TB);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN GO TO BUR;
        REPORT_NO = SUBSTR(LONG,1,N-1);
        L_REPORT_NO = N - 1;
```

```
         IF (N-1) > ((N60-59)*132/2) THEN GO TO OFF_LNTH:
BUR:     LONG = SUBSTR(LONG,N+1);


   /*  CHECK FOR A BUREAU NUMBER.  */


         N =  INDEX(LONG,CR);
         IF N = 0 THEN GO TO MISSCR;
         IF N = 1 THEN GO TO AVL_SECT;
         BUREAU_NO = SUBSTR(LONG,1,N-1);
         L_BUREAU_NO = N - 1;
         IF (N-1) > ((N60-59)*132/2) THEN GO TO OFF_LNTH;


   /*  CHECK FOR AN AVAILABILITY STRING. */


 AVL_SECT:
         L_AVAILABILITY = 0:
         CALL SEPERATE (70,N70,7,JC_SECT,LONG);
         N =  INDEX(LONG,CR);
         IF N = 0 THEN GO TO MISSCR;
         AVAILAPILITY = SUBSTR(LONG,1,N-1);
         L_AVAILABILITY = N - 1;


   /*  CHECK FOR A JOURNAL CITATION.  */


 JC_SECT:
         L_JOUR_CIT = 0:
         CALL SEPERATE (80,N80,8,PUB_SECT,LONG);
         N =  INDEX(LONG,CR);
         IF N = 0 THEN GO TO MISSCR;
         JOUR_CIT = SUBSTR(LONG,1,N-1);
         L_JOUR_CIT = N - 1;


   /*   GET THE PUBLICATION DATE FROM LINE 90.  IT ALWAYS HAS A LENGTH  OF
         7.  */


 PUB_SECT:
         L_PUB_DATE,L_PAGE,L_DESC_NOTE = 0;
         CALL SEPERATE (90,N90,9,DESC_SECT,LONG);
         N = INDEX(LONG,TB);
         IF N = 0 THEN GO TO MISSCR;
         IF N = 1 THEN DO;
                        PUB_DATE = '       ';
                        L_PUB_DATE = 7;
                        GO TO GP;
                        END;
         PUB_DATE = SUBSTR(LONG,1,N-1);
         L_PUB_DATE = N - 1;
         IF (N-1) ¬= 7 THEN GO TO OFF_LNTH;
 GP:     LONG = SUBSTR(LONG,N + 1);


   /*   GET THE PAGE NUMBER FROM LINE 90.  */
```

```
        N = INDEX(LONG,TB);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN DO;
                      PAGE = '   ';
                      L_PAGE = 3;
                      GO TO DS;
                      END;
        PAGE = SUBSTR(LONG,1,N-1);
        L_PAGE = N - 1;
        IF (N-1) > 16 THEN GO TO OFF_LNTH;
DS:     LONG = SUBSTR(LONG,N+1);


/*   CHECK FOR THE DESCRIPTIVE NOTE.  */


        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        IF N = 1 THEN GO TO DESC_SECT;
        DESC_NOTE=SUBSTR(LONG,1,N-1);
        L_DESC_NOTE = N - 1;


/*   GET THE DESCRIPTOR STRING.  */


DESC_SECT:
        L_DESCRIP = 0;
        IND = 10;
        CALL SEPERATE (100,N100,10,ID_SECT,DESCRIP);
        NO_DES = 0;
RET2:   N = INDEX(DESCRIP,CR);
        IF N = 0 THEN GO TO MISSCR;
        DESCRIP = SUBSTR(DESCRIP,1,N-1);
        L_DESCRIP = N - 1;
        IF NO_DES = 1 THEN GO TO SEARCH;


/*   CHECK FOR THE EXISTANCE OF IDENTIFIERS.  */


ID_SECT:
        L_IDENT = 0;
        IND = 11;
        IMP = 0;
        CALL SEPERATE (200,N200,11,SEARCH,LONG);
        IMP = 1;
RETRN:  IF LONG = '' THEN GO TO SEARCH;
        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        IDENT = SUBSTR(LONG,1,N-1);
        L_IDENT = N - 1;
        GO TO SEARCH;


/*   ERROR ROUTINE FOR HAVING A MISMATCHED CARD NUMBER ON THE DATA
     CARD.  */


ERR:    DO;
        PUT FILE (ERRE) EDIT
          ('MISMATCHED CARD NUMBER IN DOCUMENT ',VT)
          (SKIP(2),A(35),A(10));
        GO TO SKIP;
        END ERR;


  /*   ERROR ROUTINE FOR A MISSING TAB OR END OF LINE CHARACTER ON A DATA
       CARD.  */


MISSCR: DO;
        PUT FILE (ERRE) EDIT
          ('MISSING END OF LINE OR TAB CHARACTER IN ',VT)
          (SKIP(2),A(37),A(10));
        IF IMP = 2 THEN GO TO SKIP;
        IF IMP=1 THEN READ FILE (CARDS) IGNORE (1);
        GO TO START;
        END;


  /*   ERROR ROUTINE FOR EXCEEDING A STRING LENGTH OR DIMENSION.        */


OFF_LNTH: DO:
          PUT FILE (ERRE) EDIT
            ('LENGTH OF VARIABLE OR DIMENSION IS OFF IN ',VT)
            (SKIP(2),A(42),A(10));
          GO TO SKIP:
          END:


  /*   ERROR ROUTINE FOR MISMATCHED VT NUMBER ON DATA CARDS. IF THE MISMA
       DUE TO A BEGIN DOCUMENT CARD, NO ERROR IS FLAGGED.  */


ERROR:  IF SUBSTR(VT_NO,1,1) ¬= BD THEN GO TO BAD;
        IF (IND = 10) & (VT_NO ¬= VT) THEN DO;
                                            IMP = 0;
                                            L_IDENT = 0;
                                            NO_DES = 1;
                                            GO TO RET2;
                                            END;
        IF (IND = 11) & (VT_NO ¬= VT) THEN GO TO RETRN;
        BAD:   PUT FILE (ERRE) SKIP(2);
               PUT FILE (ERRE) EDIT
                  ('DOCUMENT NO. DOES NOT MATCH ON ONE CARD IN ',VT)
                    (SKIP,A(43),A(10));
               GO TO SKIP;


  /*   ERROR ROUTINE FOR MISSING INFORMATION THAT IS REQUIRED FOR EACH
       DOCUMENT.  TITLE, PUBLICATION DATE, ETC. ARE SUCH REQUIRED
       INFORMATION.  */


NO_INF: PUT FILE (ERRE) EDIT
          ('EXPECTED INFORMATION MISSING FROM DOCUMENT ',VT)
          (SKIP(2),A(34),A(10));


               /* THESE STATEMENTS SKIP TO THE BEGINNING OF THE
                  NEXT DOCUMENT.                                   */
```

```
              SKIP: READ FILE (CARDS) INTO (STR);
                    GET STRING (STR) EDIT (BEG)  (A(1));
                    IF BEG = BD THEN GO TO START;
                    GO TO SKIP;




/*   SEARCH FOR A DOCUMENT'S ED NUMBER AND RESUME PAGE. IF NO CARD IS F
     FOR THE DOCUMENT, AN ERROR IS RAISED.  */


SEARCH: DO I = 1 TO K;
           IF V(I) = VT THEN DO;
                             ED_NO = E(I);
                             PAGE_NO = PG(I);
                             GO TO PGE;
                             END;
        END SEARCH;
        PUT FILE (ERRE) EDIT ('NO MATCH MADE ON ED NUMBER FOR ',VT)
          (A(31),A(10));
        IF IMP=1 THEN READ FILE (CARDS) IGNORE (1);
        GO TO START;


/*   WRITE THE DOCUMENT'S DATA ON MASTER IN ITS OWN RECORD.  LIST THE R
     ON THE PRINTER AS WELL. KEEP A RECORD OF THE NUMBER OF RECORDS ON
     START PROCESSING OF NEXT DOCUMENT.  */


PGE:    L = NO_AUTHOR + NO_CORP_AUTHOR + NO_SPONSOR + 17;
        PUT STRING (CAT) EDIT (VT,CODE_NO,ED_NO,SEASON,YEAR,PAGE_NO,(
          AUTHOR(I) DO I = 1 TO NO_AUTHOR),TITLE,(CORP_AUTHOR(I) DO I =
          1 TO NO_CORP_AUTHOR),(SPONSOR_AG(I) DO I = 1 TO NO_SPONSOR),
          PRICE,CONTRACT_NO,GRANT_NO,PA_NO,REPORT_NO,BUREAU_NO,
          AVAILABILITY,JOUR_CIT,PUB_DATE,PAGE,DESC_NOTE,DESCRIP,IDENT,
          SEP,NO_AUTHOR,(L_AUTHOR(I) DO I=1 TO NO_AUTHOR),L_TITLE,
          NO_CORP_AUTHOR,(L_CORP_AUTHOR(I) DO I=1 TO NO_CORP_AUTHOR),
          NO_SPONSOR,(L_SPONSOR_AG(I) DO I=1 TO NO_SPONSOR),L_PRICE,
          L_CONTRACT_NO,L_GRANT_NO,L_PA_NO,L_REPORT_NO,L_BUREAU_NO,
          L_AVAILABILITY,L_JOUR_CIT,L_PUB_DATE,L_PAGE,L_DESC_NOTE,
          L_DESCRIP,L_IDENT)
          (A(10),A(17),A(20),A(3),A(4),A(4),(NO_AUTHOR) A,A(L_TITLE),
          (NO_CORP_AUTHOR) A,(NO_SPONSOR) A,A(L_PRICE),A(L_CONTRACT_NO)
          ,A(L_GRANT_NO),A(L_PA_NO),A(L_REPORT_NO),A(L_BUREAU_NO),
          A(L_AVAILABILITY),A(L_JOUR_CIT),
          A(L_PUB_DATE),A(L_PAGE),A(L_DESC_NOTE),A(L_DESCRIP),
          A(L_IDENT),A(2),(L) F(4));
        WRITE FILE (MASTER) FROM (CAT);
        PUT FILE (SYSPRINT) SKIP(2);
        PUT FILE (SYSPRINT) EDIT (CAT) (A(LENGTH(CAT)));
        NO_RECS = NO_RECS + 1;
        IF IMP=1 THEN READ FILE (CARDS) IGNORE (1);
        GO TO START;


 /*   THIS PROCEDURE READS DATA CARDS STARTING AT LINE -K- AND CONTINUES
      THE LINE NUMBER IS UNDER -L-.  IT PUTS THE DATA FROM THE SECTION I
      STRING -BIG-, WHICH IS RETURNED TO THE MAIN PROGRAM TO BE PROCESSE
      TESTED FOR ERRORS.  */
```

```
SEPERATE: PROCEDURE (K,L,IND,LBL_NXT,BIG);
        DECLARE
          K
              FIXED BINARY (31,0),
          L
              FIXED BINARY (31,0),
          I
              FIXED BINARY (31,0),
          J
              FIXED BINARY (31,0),
          BIG                 CHARACTER (660) VARYING,
          LITTLE
              CHARACTER (66),
          IND
              FIXED BINARY (31,0),
          LBL_NXT
              LABEL (CORP_SECT,SPON_SECT,PRICE_SECT,REP_BUR_SECT,
                     AVL_SECT,JC_SECT,PUB_SECT,DESC_SECT,ID_SECT,
                     SEARCH,TLE_SECT),
          ;
        BIG = '';
LOOP:   DO I = K TO L;
        IF TAB = 1 THEN GO TO LN_CK;
        READ FILE (CARDS) INTO (STR);
        GET STRING (STR) EDIT (VT_NO,LINE_NO,CARD_NO)
          (A(10),F(3),F(1));
        IF VT_NO ¬= VT THEN GO TO ERROR;
LN_CK:  IF LINE_NO = I THEN GO TO CD_CK;
        IF (IND=2)|(IND=9)|(IND=10)|(IND=5)
                                   THEN DO;
                                        IF I = K THEN GO TO NO_INF;
                                        TAB = 1;
                                        RETURN;
                                        END;
                                   ELSE DO;
                                        TAB = 1;
                                        IF I = K THEN GO TO LBL_NXT;
                                        RETURN;
                                        END;
CD_CK:  DO J = 1 TO 2;
        IF J = 1 THEN GO TO NO_RD;
        READ FILE (CARDS) INTO (STR);
        GET STRING (STR) EDIT (VT_NO,LINE_NO,CARD_NO)
          (A(10),F(3),F(1));
        IF VT_NO ¬= VT THEN GO TO ERROR;
        IF LINE_NO ¬= I THEN DO;
                             TAB = 1;
                             RETURN;
                             END;
NO_RD:  IF CARD_NO ¬= J THEN GO TO ERR;
        GET STRING (STR) EDIT (LITTLE) (X(14),A(66));
        BIG = BIG || LITTLE;
        END CD_CK;
        TAB = 0;
END_LP: END LOOP;
        RETURN;
        END SEPERATE;

/*   PRINT OUT THE NUMBER OF RECORDS ON MASTER. CLOSE ALL FILES.  */
```

```
STOP:    PUT DATA (NO_RECS);
         IF NO_RECS ¬= K THEN PUT FILE (ERRE) EDIT
           ('NOT ALL DOCUMENTS WERE PROCESSED. THERE SHOULD BE ',K)
           (SKIP(2),A(50),F(4));
         CLOSE FILE (MASTER), FILE (ERRE);
         CLOSE FILE (CARDS);
         END WORK;
         END BEGIN;
```

ABSTRACT

```
      /*
               PURPOSE:  READ ABSTRACT DATA CARDS AND CREATE AN ABSTRACT R
                         FOR EACH DOCUMENT.  RECORD CONSISTS OF CLEARINGH
                         ACCESSION NUMBER, 2 DIGIT CODE TO BE USED FOR SOR
                          AND THE ABSTRACT.

      */
 GENABS: PROCEDURE OPTIONS(MAIN);
 DCL LAST CHAR(4),
           CODE CHAR(2),
     A_OR_BLNK CHAR(1),
           1 RESUME_CARD,
             2 VT CHAR(10),
             2 CARD_ID CHAR(4) INITIAL('    '),
             2 DATA CHAR(66),
            VT_CURRENT CHAR(10),
           RESUME CHAR(2000) VAR:
 ON ENDFILE(CARDS) GO TO SORT;
 GET FILE(SYSIN) EDIT(A_OR_BLNK,CODE)(A(1),X(78),A(1));
      /* * * * * * * * * * * * */
      /* * * * * * * * * * * * */
      /*                       */
      /* LOOP TO READ RESUME */
      /*    AND FORM RECORD    */
      /*                       */
      /* * * * * * * * * * * * */
   FIRST: READ FILE(CARDS) INTO(RESUME_CARD);
  IF SUBSTR(VT,3,1)¬=' '|SUBSTR(VT,7,1)¬=' ' THEN DO;
   PUT FILE(SYSPRINT) SKIP(2);
   BADVT1: PUT FILE(SYSPRINT) EDIT(VT,CARD_ID,DATA)(SKIP(1),A(10),A(4),
                A(66));
           READ FILE(CARDS) INTO(RESUME_CARD);
           IF SUBSTR(VT,1,1)='¬' THEN GO TO FIRST;
           GO TO BADVT1;
                                                       END;
   IF SUBSTR(VT,1,1)='¬' THEN GO TO FIRST;
           IF A_OR_BLNK='A' THEN I=5; ELSE I=8;
           GET STRING(DATA) EDIT(CODE)(X(I),A(2));
           READ FILE(CARDS) INTO(RESUME_CARD);
           RESUME=VT||CODE||DATA;
           VT_CURRENT=VT:
            I=INDEX(DATA,'|');
           LAST=CARD_ID;
           IF I=0 THEN GO TO AGAIN;
           RESUME=SUBSTR(RESUME,1,I+12);
           WRITE FILE(SORTIN) FROM(RESUME);
           GO TO FIRST;
       /*   READ ONE CARD OF ABSTRACT CARD DECK     */
   AGAIN: READ FILE(CARDS) INTO(RESUME_CARD);
       /*  CHECK FOR ERROR IN CLEARINGHOUSE ACCESSION NUMBER    */
  IF SUBSTR(VT,3,1)¬=' '|SUBSTR(VT,7,1)¬=' ' THEN DO;
   PUT FILE(SYSPRINT) SKIP(2);
   BADVT2: PUT FILE(SYSPRINT) EDIT(VT,CARD_ID,DATA)(SKIP(1),A(10),A(4),
                A(66));
           READ FILE(CARDS) INTO(RESUME_CARD);
           IF SUBSTR(VT,1,1)='¬' THEN GO TO FIRST;
           GO TO BADVT2;
                                                       END;
         /*   CHECK FOR DOCUMENT SEPERATOR  */
   IF SUBSTR(VT,1,1)='¬' THEN GO TO FIRST;
          I=INDEX(DATA,'|');
        /*  CHECK FOR END OF ABSTRACT */
          IF VT¬=VT_CURRENT|LAST>=CARD_ID THEN DO;
   /*    CHECK CARD FOR ERROR IN CLEARINGHOUSE ACCESSION NUMBER OR SEQU
         */
       PUT FILE(SYSPRINT) EDIT(LAST)(SKIP(2),A);
BADID: PUT FILE(SYSPRINT) EDIT(RESUME_CARD)(SKIP,A(10),A(4),A(66)):
       READ FILE(CARDS) INTO(RESUME_CARD);
       IF SUBSTR(VT,1,1)='¬' THEN GO TO FIRST;
       GO TO BADID;
       END:
        IF I=0 THEN RESUME=RESUME||DATA:
                ELSE DO;
                         RESUME=RESUME||SUBSTR(DATA,1,I):
                         WRITE FILE(SORTIN) FROM(RESUME):
                         GOTO FIRST:
                      END;
  LAST=CARD-ID;
         GO TO AGAIN;
    /* * * * * * * * * * * * */
    /* * * * * * * * * * * * */
SORT: END;
```

RESUME PRINT

```
 /*

        THIS SECTION CREATES THE RESUME PRINTOUT

 */


SECT1: PROCEDURE OPTIONS(MAIN);

DCL A_OR_BLNK CHAR(1);

 /*
       READ CONTROL CARDS
 */

GET FILE(SYSIN) DATA(A_OR_BLNK);
 /*
       START PROCESSING RECORDS FOR PRINTED SECTION
 */
DCL BUF_VT CHAR(10),
    CODE CHAR(17),
        AIM_CODE CHAR(2) DEFINED CODE POSITION(6),
        ARM_CODE CHAR(2) DEFINED CODE POSITION(9),
    BUF_ED CHAR(20),
    BUF_PUB CHAR(1),
    BUF_SEASON CHAR(6),
    RESUME_PG_NO FIXED BINARY(31,0),
    CARD CHAR(80),
    BUF2 CHAR(3678) VAR,
    IN_BUFFER CHAR(3736) VAR,
           BRAKUP
              ENTRY(CHAR(2000) VAR),
    (VT_ED_SKIP,BREAK,I,J,#_P_AUTHORS,LEN_P_AUTHOR(2),TITLE_LEN,#SA,
     #_C_AUTHORS,LEN_C_AUTHOR(5),LEN_SA(3),LEN_PRICE,LEN_CONTRACT,
     LEN_JC,LEN_DATE,LEN_PAGES,LEN_DESC_NOTE,LEN_DESC,LEN_IDENT,
    PAGE_WIDTH) FIXED BINARY(31,0),
     (SPACE_LINES,INDENT) FIXED BINARY(31,0),
    (FIRST_LINE,TOP_LINE,CON_1ST_LINE) FIXED BINARY(31,0),
    RESUME CHAR(2000) VAR,
    (VT_COL,PRICE_COL,CONTRACT_COL,GRANT_COL,PA_COL) FIXED BINARY(15),
    TITLE CHAR(396) VAR,
    (REPORT_COL,BUREAU_COL,LEN_HEADING) FIXED BINARY(15,0),
    INDEX_PAGE FIXED BINARY(31,0),
    INDEX_PAGE_CHACTER CHAR(4),
    BLANKS CHAR(132) INITIAL( (132)' ' ),
   (LINE_50, LINE_60, LINE_90) CHAR(198) VAR,
    CURRENT_CODE CHAR(2) INITIAL('  '),
    OLD_CODE CHAR(2),
    HEADING CHAR(20) VAR,
    (TOTAL_LINES,TOTAL_COLUMNS) FIXED BINARY(31,0), /* OUTPUT DIMEN.*/
    UNDER CHAR(10) INITIAL( (10) '_'),
     INITIALS CHAR(4) INITIAL('    '),
    TOP CHAR(4);
    DCL TEMP_STRING CHAR(2000) VAR;
    DCL VT_OUT(20) CHAR(10),
    RES_CODE CHAR(2),
        IVT FIXED BINARY(15,0);

 /*
        EOF CONTROL
 */

```

```
ON ENDFILE(MASTER)  BEGIN:
                    SIGNAL ENDPAGE(OUT);
                    GOTO FINISH;
                    END:
ON ENDFILE(SORTOUT) BEGIN:
                    PUT FILE(SYSPRINT) LIST('RESUME HAS FEWER RECORDS');
                    GO TO FINISH;
                    END:
ON ENDPAGE(OUT) BEGIN:
      CURRENT_CODE=OLD_CODE;
       IF TOP='LEFT' THEN DO;
                           TOP='RITE';
                           J1=INDENT;
                           J2=LEN_HEADING+INDENT;
                           END:
                      ELSE DO;
                           TOP='LEFT';
                           J1=LEN_HEADING+INDENT+10;
                           J2=INDENT;
                           END;
     PUT FILE(OUT) PAGE LINE(TOP_LINE) EDIT(INDEX_PAGE,HEADING)
               (COLUMN(J1),F(4),SKIP(0),COLUMN(J2),A);
         PUT FILE(OUT) SKIP(2);
                                    TOP_LINE=CON_1ST_LINE:
        INDEX_PAGE=INDEX_PAGE+1;
                 END;
        GET FILE(SYSIN) DATA COPY:
     GET FILE(BADVT) EDIT(IVT,(VT_OUT(J) DO J=1 TO IVT))(F(4),20 A(10)):
     KEY=0;
 OPEN FILE(OUT ) PRINT PAGESIZE(TOTAL_LINES);
     LEN_HEADING=PAGE_WIDTH-LENGTH(HEADING);
                                    TOP_LINE=FIRST_LINE;

  /*
           READ BIBLIOGRAPHIC RECORD
  */

 LOOP: READ FILE(MASTER) INTO(IN_BUFFER);
       GET STRING(IN_BUFFER) EDIT(BUF_VT,CODE,BUF_ED,BUF_PUB,BUF_SEASON,
          RESUME_PG_NO,BUF2)(A(10),A(17),A(20),A(1),A(6),F(4),
          A(LENGTH(IN_BUFFER)-58));

  /*
              CHECK IF CATEGORY OR FIELD HAS CHANGED
                     IF SO,
                           PUT LAST PAGE ON PRINT FILE
                           AND SET UP NEW PAGE
                           THEN READ ABSTRACT RECORD
                     IF NOT, READ ABSTRACT RECORD
  */

       IF BUF_PUB='A'&AIM_CODE¬=CURRENT_CODE THEN DO:
                         IF TOP='LEFT'  THEN TOP='RITE';
                                        ELSE TOP='LEFT';
                                    INDEX_PAGE=INDEX_PAGE+1;
                                    OLD_CODE=AIM_CODE:
                                    SIGNAL ENDPAGE(OUT);
                                    GO TO READ_RESUME;   END;
       IF BUF_PUB¬='A'&ARM_CODE¬=CURRENT_CODE THEN DO;
                         IF TOP='LEFT'  THEN TOP='RITE';
                                        ELSE TOP='LEFT';
                                    INDEX_PAGE=INDEX_PAGE+1;
```

```
                                  OLD_CODE=ARM_CODE;
                                  SIGNAL ENDPAGE(OUT);
                                END;
       IF KEY=1 THEN GO TO NO_RES:

 /*
             READ ABSTRACT RECORD
 */

  READ_RESUME:
     READ FILE(SORTOUT) INTO(RESUME);
NO_RES: KEY=0;
    RES_CODE=SUBSTR(RESUME,11,2);
    IF RES_CODE<'00' THEN GO TO READ_RESUME;
   DO I=1 TO IVT;
     IF BUF_VT=VT_OUT(I) THEN GO TO LOOP;
   END;
     IF SUBSTR(RESUME,1,10)>BUF_VT&(CURRENT_CODE=AIM_CODE|CURRENT_CODE=
        ARM_CODE) THEN DO;
         PUT FILE(SYSPRINT) LIST('**ERROR ON MATCH** ',BUF_VT,RESUME);
     IF RES_CODE¬=CURRENT_CODE THEN DO; KEY=1;
                                        GO TO LOOP;
                                    END;
                END;

  /*
         CHECK IF ABSTRACT VT=BIBLIOGRAPHIC VT
               IF YES, CONTINUE
               IF NOT, RETURN TO BIBLIOGRAPHIC RECORD READ-IN
  */

      IF SUBSTR(RESUME,1,10)¬=BUF_VT THEN DO;
    RESUME=SUBSTR(RESUME,1,10)||BUF_VT:
   PUT FILE(SYSPRINT) LIST('** ERRORS IN VTS RESUME BIB **',RESUME);
         GO TO READ_RESUME;
                                              END:
      RESUME=SUBSTR(RESUME,13);
      I=LINENO(OUT);
 IF I>TOTAL_LINES-4 THEN DO:
                  SIGNAL ENDPAGE(OUT);
                  I=LINENO(OUT);
                                  END;
        INDEX_PAGE=INDEX_PAGE-1;
        CARD=' ';

   /*
             PUT RESUME PAGE INTO BIB RECORD AND WRITE RECORD
   */

      PUT STRING(CARD) EDIT(BUF_VT,BUF_ED,INDEX_PAGE)(A(10),A(20),F(4));
  SUBSTR(IN_BUFFER,55,4)=SUBSTR(CARD,31,4);
  WRITE FILE(NEWMAS) FROM(IN_BUFFER);

   /*
          PUNCH VT-ED RESUME PAGE CARD
   */

        WRITE FILE(PUNCH) FROM(CARD);
        INDEX_PAGE=INDEX_PAGE+1;
       IF I>TOP_LINE+3 THEN I=3; ELSE I=2;

   /*
```

```
          PUT VT-ED LINE INTO RESUME RECORD
   */

 PUT FILE(OUT) SKIP(I) EDIT(BUF_VT,UNDER,BUF_ED)(COLUMN(VT_COL),A,SKIP(0
 ),COLUMN(VT_COL),A,X(VT_ED_SKIP),A);
      I=INDEX(BUF2,'||')+1;
         LEN_P_AUTHOR=0; LEN_C_AUTHOR=0; LEN_SA=0;
         GET STRING(BUF2) EDIT(#_P_AUTHORS,(LEN_P_AUTHOR(J) DO J=1 TO
                              #_P_AUTHORS WHILE(#_P_AUTHORS>0)),
                              TITLE_LEN,
                              #_C_AUTHORS,(LEN_C_AUTHOR(J) DO J=1 TO
                              #_C_AUTHORS WHILE(#_C_AUTHORS>0)),
                            #SA,(LEN_SA(J) DO J=1 TO #SA WHILE(#SA>0)),
                              LEN_PRICE,LEN_CONTRACT,LEN_GRANT,LEN_PA,
                              LEN_REPORT,LEN_BUREAU,
                              LEN_AVAIL,
                              LEN_JC,
                              LEN_DATE,LEN_PAGES,LEN_DESC_NOTE,
                              LEN_DESC,
                              LEN_IDENT)
                             (X(I),28 F(4));
          I=1;

   /*
                 SET UP PERSONAL AUTHOR LINE
   */

         IF #_P_AUTHORS>0 THEN DO:
              IF #_P_AUTHORS=1 THEN CALL BRAKUP(SUBSTR(BUF2,1,
               LEN_P_AUTHOR(1))||' %@!');
                              ELSE CALL BRAKUP(SUBSTR(BUF2,1,
                  LEN_P_AUTHOR(1))||'; '||SUBSTR(BUF2,LEN_P_AUTHOR(1)+
               1,LEN_P_AUTHOR(2))||' %@!');
              I=I+SUM(LEN_P_AUTHOR);
                              END;
          TITLE=SUBSTR(BUF2,I,TITLE_LEN);
          J=INDEX(TITLE,'<');

   /*
                 SET UP TITLE LINE
   */

  TWO_PART_TITLE: IF J=0 THEN GO TO TITLE_PROCESS;
                  IF J=1 THEN TITLE=SUBSTR(TITLE,2);
                      ELSE TITLE=SUBSTR(TITLE,1,J-1)||SUBSTR(TITLE,J+1
                         ):
          J=INDEX(TITLE,'>');
          IF J=0 THEN DO;
               PUT FILE(SYSPRINT) LIST('TITLE ERROR',IN_BUFFER);
               GO TO LOOP;
                    END;
          SUBSTR(TITLE,J)=SUBSTR(TITLE,J+2);
          J=INDEX(TITLE,'<');
           GO TO TWO_PART_TITLE;
  TITLE_PROCESS: CALL BRAKUP(TITLE||' %@!');
          PUT FILE(OUT) SKIP(SPACE_LINES);
          I=I+TITLE_LEN;

    /*
                 SET UP CORPORATE AUTHOR LINE
    */
```

```
     IF #_C_AUTHORS>0 THEN DO J=1 TO #_C_AUTHORS;
            CALL BRAKUP(SUBSTR(BUF2,I,LEN_C_AUTHOR(J))||' %@!');
            I=I+LEN_C_AUTHOR(J);
                                 END:
      IF #SA>0 THEN DO J=1 TO #SA;
             CALL BRAKUP(SUBSTR(BUF2,I,LEN_SA(J))||' %@!');
            I=I+LEN_SA(J);
                    END;

/*
              SET UP PRICE LINE
*/

   BEGIN;
      IF LEN_PRICE>0 THEN DO;
   LINE_50=SUBSTR(BLANKS,1,PRICE_COL-1);
   LINE_50=LINE_50||SUBSTR(BUF2,I,LEN_PRICE)||SUBSTR(BLANKS,1,
        CONTRACT_COL);
            I=I+LEN_PRICE;
   IF LEN_CONTRACT=0 THEN LINE_50=LINE_50||SUBSTR(BUF2,I,LEN_GRANT);
                     ELSE LINE_50=LINE_50||SUBSTR(BUF2,I,LEN_CONTRACT)
                      :
            I=I+LEN_CONTRACT;
            I=I+LEN_GRANT;
   IF LEN_PA¬=0 THEN LINE_50=LINE_50||SUBSTR(BLANKS,1,PA_COL)||
        SUBSTR(BUF2,I,LEN_PA);
            I=I+LEN_PA;
   PUT FILE(OUT) EDIT(LINE_50)(SKIP,A);
                          END;
   END:

/*
              SET UP REPORT AND BUREAU NUMBER LINES
*/

   BEGIN;
      IF LEN_REPORT>0|LEN_BUREAU>0 THEN DO;
   LINE_60=SUBSTR(BLANKS,1,PRICE_COL-1);
IF LEN_REPORT¬=0 THEN
   LINE_60=LINE_60||SUBSTR(BUF2,I,LEN_REPORT)||SUBSTR(BLANKS,1,
        LEN_PRICE-LEN_REPORT);
                  ELSE
   LINE_60=LINE_60||SUBSTR(BLANKS,1,LEN_PRICE);
         I=I+LEN_REPORT:
   IF LEN_BUREAU¬=0 THEN
LINE_60=LINE_60||SUBSTR(BLANKS,1,GRANT_COL)||SUBSTR(BUF2,I,
     LEN_BUREAU);
          I=I+LEN_BUREAU:
   PUT FILE(OUT) EDIT(LINE_60)(SKIP,A);
                                         END;
   END;

/*
               SET UP AVAILABILITY LINE
*/

    IF LEN_AVAIL>0 THEN CALL BRAKUP(SUBSTR(BUF2,I,LEN_AVAIL)'|' %@!');
             I=I+LEN_AVAIL;
       IF LEN_JC>0 THEN CALL BRAKUP(SUBSTR(BUF2,I,LEN_JC)||' %@!');
             I=I+LEN_JC:
    BEGIN;
       PUT STRING(LINE_90) EDIT('PUB DATE - ',SUBSTR(BUF2,I,LEN_DATE),
             '  ',SUBSTR(BUF2,I+LEN_DATE,LEN_PAGES))(A(11),A,A(2),A);
             IF SUBSTR(LINE_90,LEN_DATE+9,2)='  ' THEN SUBSTR(LINE_90,
                   LEN_DATE+9,2)='ND';
             IF LEN_DESC_NOTE>0 THEN LINE_90=LINE_90||'; '||SUBSTR(BUF2
                   ,I+LEN_DATE+LEN_PAGES,LEN_DESC_NOTE);
             I=I+LEN_DATE+LEN_PAGES+LEN_DESC_NOTE;
              CALL BRAKUP(LINE_90||' %@!');
       PUT FILE(OUT) SKIP(SPACE_LINES);
    END;

/*
               SET UP DESCRIPTOR AND IDENTIFIER LINES
*/

    IF LEN_DESC>0 THEN DO;
         TEMP_STRING='DESCRIPTORS - '||SUBSTR(BUF2,I,LEN_DESC);
              TEMP_STRING=TEMP_STRING||' %@!';
         CALL BRAKUP(TEMP_STRING);
             I=I+LEN_DESC;
                        END;
    IF LEN_IDENT>0 THEN DO;
         TEMP_STRING='IDENTIFIERS - '||SUBSTR(BUF2,I,LEN_IDENT);
              TEMP_STRING=TEMP_STRING||' %@!';
         CALL BRAKUP(TEMP_STRING);
                        END;

/*
               SET UP ABSTRACT LINE
*/

       I= INDEX(RESUME,'|');
      IF I=0 THEN I=LENGTH(RESUME)+1;
       INITIALS=SUBSTR(RESUME,I-5,4);
       RESUME=SUBSTR(RESUME,1,I-1);
              RESUME=RESUME||' %@!';
       PUT FILE(OUT) SKIP(SPACE_LINES);
         TEMP_STRING='ABSTRACT - '||RESUME;
         CALL BRAKUP(TEMP_STRING);
       INITIALS='    ';
       GO TO LOOP; /* START NEXT DOCUMENT*/
FINISH: BEGIN;
    CLOSE FILE(OUT), FILE(SORTOUT), FILE(MASTER);
        PUT FILE(SYSPRINT) DATA(INDEX_PAGE);
END FINISH;


 /*
            BRAKUP IS USED BY THE MAIN PROGRAM IN BREAKING UP
                   THE BIBLIOGRAPHIC RECORD FOR THE RESUME.
                   RESUME RECORDS ARE PRINTED AND AT A PAGE'S
                   END THE TOP OF THE NEXT PAGE IS SET UP.
 */

BRAKUP: PROCEDURE(LINE);
DCL LINE CHAR(2000) VAR,
    OUT_LINE CHAR(132) VAR,
CHECK_INDEX FIXED BINARY(31,0),
    (J,I,K) FIXED BINARY(15,0);
    J=PAGE_WIDTH;
    OUT_LINE=LINE;
        IF LENGTH(LINE)<PAGE_WIDTH THEN DO;
        I=INDEX(OUT_LINE,' %@!');
```

```
           OUT_LINE=SUBSTR(OUT_LINE,1,I);
       PUT FILE(OUT) EDIT(OUT_LINE)(SKIP,X(INDENT),A);
             RETURN;
                                        END;
KLOOP: DO K=1 TO 50;
      IF J-PAGE_WIDTH>=LENGTH(LINE) THEN RETURN;
ILOOP: DO I=0 TO PAGE_WIDTH-2;
             IF SUBSTR(LINE,J-I,1)=' '|SUBSTR(LINE,J-I,1)='-' THEN DO;
             IF LENGTH(LINE)-J>=0 THEN
             OUT_LINE=SUBSTR(LINE,J-PAGE_WIDTH+1,PAGE_WIDTH-I);
                                  ELSE
          OUT_LINE=SUBSTR(LINE,J-PAGE_WIDTH+1);
       GO TO NEXT_LINE;
                                    END;
                END ILOOP;
NEXT_LINE: J=J+PAGE_WIDTH-I;
    CHECK_INDEX=INDEX(OUT_LINE,'%@!');
    IF CHECK_INDEX>0 THEN DO;
       IF CHECK_INDEX<3 THEN RETURN;
                         ELSE
          OUT_LINE=SUBSTR(OUT_LINE,1,CHECK_INDEX-1);
        PUT FILE(OUT) EDIT(OUT_LINE)(SKIP,X(INDENT),A);
          RETURN;
                           END;
       PUT FILE(OUT) EDIT(OUT_LINE)(SKIP,X(INDENT),A);
END KLOOP;
END BRAKUP;



                   END SECT1;
```

MASTER EDIT

```
/*********************************************************************/
/*                                                                   */
/*       E D I T   P R O G R A M   F O R   M A S T E R               */
/*                                                                   */
/*                                                                   */
/*  PURPOSE:  THE PROGRAM TAKES A MASTER TAPE DATA SET AND COPIES    */
/*             IT ONTO ANOTHER TAPE WHILE EDITING IT.                */
/*                                                                   */
/*  PROGRAM EXPLANATION:                                             */
/*    THE PROGRAM IS GIVEN A LIST OF DOCUMENTS WHICH HAVE ERRORS     */
/*    IN THEIR RECORDS ON MASTER.  THE PROGRAM COPIES THE DATA SET   */
/*    ONTO ANOTHER TAPE BUT THE BAD DOCUMENT RECORDS ARE LEFT OFF OF */
/*    THE NEW TAPE.  WHEN THE COPYING IS COMPLETE, THE BAD DOCUMENTS'*/
/*    DATA CARDS ARE READ IN.  OF COURSE, THE ERRORS ON THESE CARDS  */
/*    HAVE BEEN CORRECTED.  A PROCEDURE ALMOST IDENTICAL TO THE      */
/*    MASTER PROGRAM IS THEN CALLED TO PROCESS THESE DATA CARDS AND  */
/*    PUT THE DATA ON THE NEW TAPE.  SO THESE DOCUMENTS WHICH HAD    */
/*    ERRORS BEFORE ARE NOW CORRECT AND ARE AT THE END OF THE NEW    */
/*    DATA SET.                                                      */
/*                                                                   */
/*  INPUT TO PROGRAM:                                                */
/*    -SYSIN-, INPUT ON CARDS                                        */
/*      CARD 1 -- THE NUMBER OF DOCUMENTS WITH BAD RECORDS ON THE    */
/*                MASTER TAPE DATA SET.  COLS 1-4, RIGHT JUSTIFIED.  */
/*                -NA- IS THIS NUMBER.                               */
/*      CARDS 2 - (NA+1) -- THE VT NUMBERS OF THE DOCUMENTS, ONE TO  */
/*                            A CARD IN COLS 1-10.                   */
/*      CARD NA+2 -- VALUES FOR THE VARIABLES NA, N10, NT, NC, NS,   */
/*                   N50, N60, N70, N80, N90, N100, N200 ARE ON THIS */
/*                   CARD.  EACH IS RIGHT JUSTIFIED IN CONSECUTIVE   */
/*                   3 COLUMN FIELD WIDTHS.  SEE THE MASTER PROGRAM  */
/*                   DOCUMENTATION FOR THEIR VALUES.                 */
/*      CARD NA+3 -- THE EXPECTED NUMBER OF DOCUMENTS TO BE PROCESSED*/
/*                   THIS RUN.  COLS. 1-5, RIGHT JUSTIFIED.          */
/*      CARD NA+4 -- COLUMN 1 HAS 'A' FOR AIM OR ' ' FOR ARM.        */
/*                   COLUMNS 2-3 HAS A SEASON, 'F ','W ','SP', OR    */
/*                     'SU'.                                         */
/*                   COLUMS  4-7 IS THE CURRENT YEAR, 19__.          */
/*      CARDS NA+5 - ? -- THE VT#, ED#, PAGE# CARDS,WITH A BLANK     */
/*                        CARD AT THE END OF THE DECK,FOLLOW HERE.   */
/*                        THESE CARDS ARE FOR THE BAD DOCUMENTS ONLY.*/
/*    -CARDS-                                                        */
/*      THESE ARE THE DATA CARDS FOR THE DOCUMENTS WITH ERRORS WHICH */
/*      HAVE BEEN CORRECTED.  THE DECK STARTS AND ENDS WITH A BEGIN  */
/*      DOCUMENT CARD ('¬' IN COLUMN 1, BLANK IN COLS. 2-80) AND SUCH*/
/*      A CARD SEPARATES THE DATA FOR EACH DOCUMENT.                 */
/*                                                                   */
/*********************************************************************/

EDIT:   PROCEDURE OPTIONS (MAIN);
        DECLARE
         CAT                 CHARACTER (3736) VARYING,
         DUMMY
              CHARACTER (1),
         (NA,N10,NT,NC,NS,N50,N60,N70,N80,N90,N100,N200)
              FIXED BINARY (31,0),
           NDOC              FIXED BINARY (31,0),
           MASTI             FILE SEQUENTIAL INPUT RECORD,
          MASTER               FILE SEQUENTIAL OUTPUT RECORD,
          V(900)               CHARACTER (10),
          VT
              CHARACTER (10),
        ;

/*  COPY THE OLD MASTER DATA SET TO A NEW MASTER DATA SET, LEAVING    */
/*  OFF THE DOCUMENTS WHOSE RECORDS HAVE ERRORS.                      */


        GET FILE (SYSIN) EDIT (NA,DUMMY) (F(4),X(75),A(1));
LOOP1:  DO NC = 1 TO NA;
        GET FILE (SYSIN) EDIT (V(NC),DUMMY) (A(10),X(69),A(1));
        END LOOP1;
        ON ENDFILE (MAST1) GO TO DATA;
READ:   READ FILE (MAST1) INTO (CAT);
        GET STRING (CAT) EDIT (VT) (A(10));
LOOP2:  DO NC = 1 TO NA;
        IF VT = V(NC) THEN GO TO READ;
        END LOOP2;
        WRITE FILE (MASTER) FROM (CAT);
        PUT LIST (VT);
        GO TO READ;
DATA:   CLOSE FILE (MAST1);
        GET FILE (SYSIN) EDIT (NA,N10,NT,NC,NS,N50,N60,N70,N80,N90,
            N100,N200,DUMMY) (12 F(3),X(43),A(1))
        GET FILE (SYSIN) EDIT (NDOC,DUMMY) (F(5),X(74),A(1));

/*  THIS BLOCK IS ALMOST IDENTICAL TO THE MASTER PROGRAM.  IT WILL    */
/*  PROCESS DATA CARDS FOR A DOCUMENT AND WRITE A RECORD ON THE       */
/*  NEW MASTER DATA SET FOR THAT DOCUMENT.                            */


WORK:   BEGIN;
        DECLARE
          AUTHOR(NA)           CHARACTER (50) VARYING,
          AVAILABILITY         CHARACTER ((N70-69)*132) VARYING,
          BD                   CHARACTER (1) INITIAL ('¬'),
          BEG
              CHARACTER (1),
          BT                   CHARACTER (1) INITIAL ('<'),
          BUREAU_NO            CHARACTER ((N60-59)*132/2) VARYING,
          CARD_NO
              FIXED BINARY (31,0),
          CODE_NO
              CHARACTER (17),
          CONTRACT_NO          CHARACTER ((N50-49)*132/3) VARYING,
          CORP_AUTHOR(NC-29) CHARACTER (132) VARYING,
          CR                   CHARACTER (1) INITIAL ('|'),
          DESCRIP              CHARACTER ((N100-99)*132) VARYING,
          DESC_NOTE            CHARACTER ((N90-89)*132-13) VARYING,
          F(NDOC)              CHARACTER (20),
          ED_NO                CHARACTER (20),
          ET                   CHARACTER (1) INITIAL ('>'),
GRANT_NO            CHARACTER ((N50-49)*132/3) VARYING,
I
     FIXED BINARY (31,0),
IDENT               CHARACTER ((N200-199)*132) VARYING,
IND                 FIXED BINARY (31,0),
JOUR_CIT            CHARACTER ((N80-79)*132) VARYING,
K
     FIXED BINARY (31,0),
L
     FIXED BINARY (31,0),
LINE_NO
     FIXED BINARY (31,0),
LONG                CHARACTER (660) VARYING,
L_AUTHOR(NA)        FIXED BINARY (31,0),
L_AVAILABILITY
     FIXED BINARY (31,0),
L_BUREAU_NO
     FIXED BINARY (31,0),
L_CONTRACT_NO
     FIXED BINARY (31,0),
L_CORP_AUTHOR(NC-29) FIXED BINARY (31,0),
L_DESCRIP
     FIXED BINARY (31,0),
L_DESC_NOTE
     FIXED BINARY (31,0),
L_GRANT_NO
     FIXED BINARY (31,0),
L_IDENT
     FIXED BINARY (31,0),
L_JOUR_CIT
     FIXED BINARY (31,0),
L_PAGE
     FIXED BINARY (31,0),
L_PA_NO
     FIXED BINARY (31,0),
L_PRICE
     FIXED BINARY (31,0),
L_PUB_DATE
     FIXED BINARY (31,0),
L_REPORT_NO
     FIXED BINARY (31,0),
L_SPONSOR_AG(NS-39) FIXED BINARY (31,0),
L_TITLE
     FIXED BINARY (31,0),
N
     FIXED BINARY (31,0),
NO_AUTHOR
     FIXED BINARY (31,0),
NO_CORP_AUTHOR
     FIXED BINARY (31,0),
NO_DES
     FIXED BINARY (31,0),
NO_RECS
     FIXED BINARY (31,0),
NO_SPONSOR
     FIXED BINARY (31,0),
          PAGE
              CHARACTER (16) VARYING,
          PAGE_NO             CHARACTER (4),
          PA_NO
              CHARACTER (2),
          PG(NDOC)            CHARACTER (4),
          PRICE               CHARACTER ((N50-49)*132/2) VARYING,
          PUB_DATE
              CHARACTER (7),
          REPORT_NO           CHARACTER ((N60-59)*132/2) VARYING,
          SEASON              CHARACTER (3),
          SEP                 CHARACTER (2) INITIAL ('||'),
          SEPERATE
              ENTRY (FIXED BINARY (31,0),FIXED BINARY (31,0),
                     FIXED BINARY (31,0),,),
          SPONSOR_AG(NS-39) CHARACTER (132) VARYING,
          STR                 CHARACTER (80),
          TAB
              FIXED BINARY (31,0),
          TR                  CHARACTER (1) INITIAL ('¬'),
          TITLE               CHARACTER ((NT-19)*132) VARYING,
          VT_NO
              CHARACTER (10),
          YEAR
              CHARACTER (4),
          ;
        OPEN FILE (ERRE) PRINT STREAM OUTPUT LINESIZE(125),
             FILE (CARDS) INPUT RECORD;
        ON ENDFILE (CARDS) GO TO STOP;
        ON CONVERSION BEGIN;
          PUT FILE (ERRE) EDIT ('CONVERSION ERROR IN DOCUMENT ',VT)
             SKIP(2),A(29),A(10));
          GO TO SKIP;
          END;
        GET FILE (SYSIN) EDIT (SEASON,YEAR,DUMMY)
            (A(3),A(4),X(72),A(1));
ED_LP:  DO I = 1 TO NDOC;
           GET FILE (SYSIN) EDIT (V(I),F(I),PG(I),DUMMY)
               (A(10),A(20),A(4),X(45),A(1));
           IF V(I) = '            ' THEN GO TO ASGN;
           K = I;
        END ED_LP;
ASGN:   NO_RECS = 0;
        READ FILE (CARDS) IGNORE (1);
START:  READ FILE (CARDS) INTO (STR);
        GET STRING (STR) EDIT (VT_NO,LINE_NO,CARD_NO,CODE_NO)
          (A(10),F(3),F(1),A(17));
        IND = 0;
        IMP = 2;
        TAB = 0;
        VT = VT_NO;
        N = INDEX(CODE_NO,CR);
        IF N = 0 THEN GO TO TEST;
        CODE_NO = SUBSTR(CODE_NO,1,N-1);
TEST:   IF LINE_NO ¬= 1 THEN GO TO ERR;
        IF CARD_NO ¬= 1 THEN GO TO ERR;
        IF (SUBSTR(CODE_NO,6,2)<40) | (SUBSTR(CODE_NO,6,2)>52)
          THEN DO;
                PUT FILE (ERRF) EDIT ('CODE NO. IS OUT OF BOUNDS')
                  (SKIP(2),A(25));
                GO TO SKIP;
                END;
        NO_AUTHOR = 0;
        CALL SEPERATE(10,N10,1,TLE_SECT,LONG);
SRCH:   N = INDEX(LONG,';');
        IF N ¬= 0 THEN DO;
                       NO_AUTHOR = NO_AUTHOR + 1;
                       IF NO_AUTHOR > NA THEN GO TO OFF_LNTH;
                       AUTHOR(NO_AUTHOR) = SUBSTR(LONG,1,N-1);
                       L_AUTHOR(NO_AUTHOR) = N - 1;
                       IF (N-1) > 50 THEN GO TO OFF_LNTH;
                       LONG = SUBSTR(LONG,N+2);
                       GO TO SRCH;
                       END;
        NO_AUTHOR = NO_AUTHOR + 1;
        IF NO_AUTHOR > NA THEN GO TO OFF_LNTH;
        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        AUTHOR(NO_AUTHOR) = SUBSTR(LONG,1,N-1);
        L_AUTHOR(NO_AUTHOR) = N - 1;
        IF (N-1) > 50 THEN GO TO OFF_LNTH;
TLE_SECT:
        CALL SEPERATE (20,NT,2,CORP_SECT,LONG);
        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        TITLE = SUBSTR(LONG,1,N-1);
        L_TITLE = N - 1;
        I = 0;
        IDENT = TITLE;
RET:    N = INDEX(IDENT,BT);
        L = INDEX(IDENT,ET);
        IF (N=0) & (L=0) & (I¬=0) THEN GO TO CORP_SECT;
        IF (N=0) | (L=0)
          THEN DO;
                PUT FILE (ERRF) EDIT
                  ('SHORT TITLE MARKS ARE NOT CORRECT IN DOCUMENT ',VT)
                  (SKIP(2),A(46),A(10));
                GO TO SKIP;
                END;
        I = I + 1;
        IDENT = SUBSTR(IDENT,L+2);
        GO TO RET;
CORP_SECT:
        NO_CORP_AUTHOR = 0;
CP_LP:  DO I = 30 TO NC;
        CALL SEPERATE (I,I,3,SPON_SECT,LONG);
        NO_CORP_AUTHOR = NO_CORP_AUTHOR + 1;
        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        CORP_AUTHOR(NO_CORP_AUTHOR) = SUBSTR(LONG,1,N-1);
        L_CORP_AUTHOR(NO_CORP_AUTHOR) = N - 1;
        IF (N-1) > 132 THEN GO TO OFF_LNTH;
```

```
          END CP_LP;
 SPON_SECT:
          NO_SPONSOR = 0;
 SP_LP:   DO I = 40 TO NS;
          CALL SEPERATE (I,I,4,PRICE_SECT,LONG);
          NO_SPONSOR=NO_SPONSOR + 1;
          N = INDEX(LONG,CR);
          IF N = 0 THEN GO TO MISSCR;
          SPONSOR_AG(NO_SPONSOR) = SUBSTR(LONG,1,N-1);
          L_SPONSOR_AG(NO_SPONSOR) = N - 1;
          IF (N-1) > 132 THEN GO TO OFF_LNTH;
          END SP_LP;
 PRICE_SECT:
          L_CONTRACT_NO,L_GRANT_NO,L_PA_NO = 0;
          CALL SEPERATE (50,N50,5,REP_BUR_SECT,LONG);
          N = INDEX(LONG,TB);
          IF N = 0 THEN GO TO MISSCR;
          IF N = 1 THEN GO TO OFF_LNTH;
          PRICE = SUBSTR(LONG,1,N-1);
          L_PRICE = N - 1;
          IF (N-1) > ((N50-49)*132/2) THEN GO TO OFF_LNTH;
          LONG = SUBSTR(LONG,N+1);
          N = INDEX(LONG,TB);
          IF N = 0 THEN GO TO MISSCR;
          IF N = 1 THEN GO TO MISS1;
          CONTRACT_NO = SUBSTR(LONG,1,N-1);
          L_CONTRACT_NO = N - 1;
          IF (N-1) > ((N50-49)*132/3) THEN GO TO OFF_LNTH;
 MISS1:   LONG = SUBSTR(LONG,N+1);
          N = INDEX(LONG,TB);
          IF N = 0 THEN GO TO MISSCR;
          IF N = 1 THEN GO TO MISS2;
          GRANT_NO = SUBSTR(LONG,1,N-1);
          L_GRANT_NO = N - 1;
          IF (N-1) > ((N50-49)*132/3) THEN GO TO OFF_LNTH;
 MISS2:   LONG = SUBSTR(LONG,N+1);
          N = INDEX(LONG,CR);
          IF N = 0 THEN GO TO MISSCR;
          IF N = 1 THEN GO TO REP_BUR_SECT;
          PA_NO = SUBSTR(LONG,1,N-1);
          L_PA_NO = N - 1;
          IF (N-1) ¬= 2 THEN GO TO OFF_LNTH;
 REP_BUR_SECT:
          L_REPORT_NO,L_BUREAU_NO = 0;
          CALL SEPERATE(60,N60,6,AVL_SECT,LONG);
          N = INDEX(LONG,TB);
          IF N = 0 THEN GO TO MISSCR;
          IF N = 1 THEN GO TO BUR;
          REPORT_NO = SUBSTR(LONG,1,N-1);
          L_REPORT_NO = N - 1;
          IF (N-1) > ((N60-59)*132/2) THEN GO TO OFF_LNTH;
 BUR:     LONG = SUBSTR(LONG,N+1);
          N = INDEX(LONG,CR);
          IF N = 0 THEN GO TO MISSCR;
          IF N = 1 THEN GO TO AVL_SECT;
          BUREAU_NO = SUBSTR(LONG,1,N-1);
```

```
         L_BUREAU_NO = N - 1;
         IF (N-1) > ((N60-59)*132/2) THEN GO TO OFF_LNTH;
 AVL_SECT:
         L_AVAILABILITY = 0;
         CALL SEPERATE (70,N70,7,JC_SECT,LONG);
         N = INDEX(LONG,CR);
         IF N = 0 THEN GO TO MISSCR;
         AVAILABILITY = SUBSTR(LONG,1,N-1);
         L_AVAILABILITY = N - 1;
 JC_SECT:
         L_JOUR_CIT = 0;
         CALL SEPERATE (80,N80,8,PUB_SECT,LONG);
         N = INDEX(LONG,CR);
         IF N = 0 THEN GO TO MISSCR;
         JOUR_CIT = SUBSTR(LONG,1,N-1);
         L_JOUR_CIT = N - 1;
 PUB_SECT:
         L_PUB_DATE,L_PAGE,L_DESC_NOTE = 0;
         CALL SEPERATE (90,N90,9,DESC_SECT,LONG);
         N = INDEX(LONG,TB);
         IF N = 0 THEN GO TO MISSCR;
         IF N = 1 THEN DO;
                         PUB_DATE = '       ';
                         L_PUB_DATE = 7;
                         GO TO GP;
                         END;
         PUB_DATE = SUBSTR(LONG,1,N-1);
         L_PUB_DATE = N - 1;
         IF (N-1) ¬= 7 THEN GO TO OFF_LNTH;
 GP:     LONG = SUBSTR(LONG,N + 1);
         N = INDEX(LONG,TB);
         IF N = 0 THEN GO TO MISSCR;
         IF N = 1 THEN DO;
                         PAGE = '   ';
                         L_PAGE = 3;
                         GO TO DS;
                         END;
         PAGE = SUBSTR(LONG,1,N-1);
         L_PAGE = N - 1;
         IF (N-1) > 16 THEN GO TO OFF_LNTH;
 DS:     LONG = SUBSTR(LONG,N+1);
         N = INDEX(LONG,CR);
         IF N = 0 THEN GO TO MISSCR;
         IF N = 1 THEN GO TO DESC_SECT;
         DESC_NOTE=SUBSTR(LONG,1,N-1);
         L_DESC_NOTE = N - 1;
 DESC_SECT:
         L_DESCRIP = 0;
         IND = 10;
         CALL SEPERATE (100,N100,10,ID_SECT,DESCRIP);
         NO_DES = 0;
 RET2:   N = INDEX(DESCRIP,CR);
         IF N = 0 THEN GO TO MISSCR;
         DESCRIP = SUBSTR(DESCRIP,1,N-1);
         L_DESCRIP = N - 1;
         IF NO_DES = 1 THEN GO TO SEARCH;
```

```
ID_SECT:
        L_IDENT = 0;
        IND = 11;
        IMP = 0;
        CALL SEPERATE (200,N200,11,SEARCH,LONG);
        IMP = 1;
RETRN:  IF LONG = '' THEN GO TO SEARCH;
        N = INDEX(LONG,CR);
        IF N = 0 THEN GO TO MISSCR;
        IDENT = SUBSTR(LONG,1,N-1);
        L_IDENT = N - 1;
        GO TO SEARCH;
ERR:    DO;
        PUT FILE (ERRE) EDIT
          ('MISMATCHED CARD NUMBER IN DOCUMENT ',VT)
          (SKIP(2),A(35),A(10));
        GO TO SKIP;
        END ERR;
 MISSCR: DO;
         PUT FILE (ERRE) EDIT
           ('MISSING END OF LINE OR TAB CHARACTER IN ',VT)
           (SKIP(2),A(37),A(10));
         IF IMP = 2 THEN GO TO SKIP;
         IF IMP=1 THEN READ FILE (CARDS) IGNORE (1);
         GO TO START;
         END;
 OFF_LNTH: DO;
           PUT FILE (ERRE) EDIT
             ('LENGTH OF VARIABLE OR DIMENSION IS OFF IN ',VT)
             (SKIP(2),A(42),A(10));
           GO TO SKIP;
           END;
 ERROR:  IF SUBSTR(VT_NO,1,1) ¬= BD THEN GO TO BAD;
         IF (IND = 10) & (VT_NO ¬= VT) THEN DO;
                                              IMP = 0;
                                              L_IDENT = 0;
                                              NO_DES = 1;
                                              GO TO RET2;
                                              END;
         IF (IND = 11) & (VT_NO ¬= VT) THEN GO TO RETRN;
         BAD:    PUT FILE (ERRE) SKIP(2);
                 PUT FILE (ERRE) EDIT
                   ('DOCUMENT NO. DOES NOT MATCH ON ONE CARD IN ',VT)
                   (A(43),A(10));
                 GO TO SKIP;
 NO_INF: PUT FILE (ERRE) EDIT
           ('EXPECTED INFORMATION MISSING FROM DOCUMENT ',VT)
           (SKIP(2),A(34),A(10));

            /* THESE STATEMENTS SKIP TO THE BEGINNING OF THE           */
               NEXT DOCUMENT.

            SKIP: READ FILE (CARDS) INTO (STR);
                  GET STRING (STR) EDIT (BEG)  (A(1));
                  IF BEG = BD THEN GO TO START;
```

```
                    GO TO SKIP;


SEARCH: DO I = 1 TO K;
           IF V(I) = VT THEN DO;
                             ED_NO = E(I);
                             PAGE_NO = PG(I);
                             GO TO PGE;
                             END;
        END SEARCH;
        PUT FILE (ERRE) EDIT ('NO MATCH MADE ON ED NUMBER FOR ',VT)
          (A(31),A(10));
        IF IMP=1 THEN READ FILE (CARDS) IGNORE (1);
        GO TO START;
PGE:    L = NO_AUTHOR + NO_CORP_AUTHOR + NO_SPONSOR + 17;
        PUT STRING (CAT) EDIT (VT,CODE_NO,ED_NO,SEASON,YEAR,PAGE_NO,(
          AUTHOR(I) DO I = 1 TO NO_AUTHOR),TITLE,(CORP_AUTHOR(I) DO I =
          1 TO NO_CORP_AUTHOR),(SPONSOR_AG(I) DO I = 1 TO NO_SPONSOR),
          PRICE,CONTRACT_NO,GRANT_NO,PA_NO,REPORT_NO,BUREAU_NO,
          AVAILABILITY,JOUR_CIT,PUB_DATE,PAGE,DESC_NOTE,DESCRIP,IDENT,
          SEP,NO_AUTHOR,(L_AUTHOR(I) DO I=1 TO NO_AUTHOR),L_TITLE,
          NO_CORP_AUTHOR,(L_CORP_AUTHOR(I) DO I=1 TO NO_CORP_AUTHOR),
          NO_SPONSOR,(L_SPONSOR_AG(I) DO I=1 TO NO_SPONSOR),L_PRICE,
          L_CONTRACT_NO,L_GRANT_NO,L_PA_NO,L_REPORT_NO,L_BUREAU_NO,
          L_AVAILABILITY,L_JOUR_CIT,L_PUB_DATE,L_PAGE,L_DESC_NOTE,
          L_DESCRIP,L_IDENT)
          (A(10),A(17),A(20),A(3),A(4),A(4),(NO_AUTHOR) A,A(L_TITLE),
          (NO_CORP_AUTHOR) A,(NO_SPONSOR) A,A(L_PRICE),A(L_CONTRACT_NO)
          ,A(L_GRANT_NO),A(L_PA_NO),A(L_REPORT_NO),A(L_BUREAU_NO),
          A(L_AVAILABILITY),A(L_JOUR_CIT),
          A(L_PUB_DATE),A(L_PAGE),A(L_DESC_NOTE),A(L_DESCRIP),
          A(L_IDENT),A(2),(L) F(4));
        WRITE FILE (MASTER) FROM (CAT);
        PUT FILE (SYSPRINT) SKIP(2);
        PUT FILE (SYSPRINT) EDIT (CAT) (A(LENGTH(CAT)));
        NO_RECS = NO_RECS + 1;
        IF IMP=1 THEN READ FILE (CARDS) IGNORE (1);
        GO TO START;
SEPERATE: PROCEDURE (K,L,IND,LBL_NXT,BIG);
        DECLARE
          K
              FIXED BINARY (31,0),
          L
              FIXED BINARY (31,0),
          I
              FIXED BINARY (31,0),
          J
              FIXED BINARY (31,0),
          BIG                  CHARACTER (660) VARYING,
          LITTLE
              CHARACTER (66),
          IND
              FIXED BINARY (31,0),
          LBL_NXT
              LABEL (CORP_SECT,SPON_SECT,PRICE_SECT,REP_BUR_SECT,
                     AVL_SECT,JC_SECT,PUB_SECT,DESC_SECT,ID_SECT,
                     SEARCH,TLE_SECT),
          ;
        BIG = '';
LOOP:   DO I = K TO L:
        IF TAB = 1 THEN GO TO LN_CK;
        READ FILE (CARDS) INTO (STR);
        GET STRING (STR) EDIT (VT_NO,LINE_NO,CARD_NO)
          (A(10),F(3),F(1));
        IF VT_NO ¬= VT THEN GO TO ERROR;
LN_CK:  IF LINE_NO = I THEN GO TO CD_CK;
        IF (IND=2)|(IND=9)|(IND=10)|(IND=5)
                                    THEN DO;
                                         IF I = K THEN GO TO NO_INF;
                                         TAB = 1;
                                         RETURN;
                                         END;
                                    ELSE DO;
                                         TAB = 1;
                                         IF I = K THEN GO TO LBL_NXT;
                                         RETURN;
                                         END;
CD_CK:  DO J = 1 TO 2;
        IF J = 1 THEN GO TO NO_RD;
        READ FILE (CARDS) INTO (STR);
        GET STRING (STR) EDIT (VT_NO,LINE_NO,CARD_NO)
          (A(10),F(3),F(1));
        IF VT_NO ¬= VT THEN GO TO ERROR;
        IF LINE_NO ¬= I THEN DO;
                             TAB = 1;
                             RETURN;
                             END;
NO_RD:  IF CARD_NO ¬= J THEN GO TO ERR;
        GET STRING (STR) EDIT (LITTLE) (X(14),A(66));
        BIG = BIG || LITTLE;
        END CD_CK;
        TAB = 0;
END_LP: END LOOP;
        RETURN;
        END SEPERATE;
STOP:   PUT DATA (NO_RECS);
        IF NO_RECS ¬= K THEN PUT FILE (ERRE) EDIT
          ('NOT ALL DOCUMENTS WERE PROCESSED. THERE SHOULD BE ',K)
          (SKIP(2),A(50),F(4));
        CLOSE FILE (MASTER), FILE (ERRE);
        CLOSE FILE (CARDS);
        END WORK;
        END EDIT;
```

SUBJECT QUARTERLY

```
/****************************************************************************/
/*                                                                         */
/*          Q U A R T E R L Y   S U B J E C T   I N D E X                  */
/*                                                                         */
/*                                                                         */
/*  PURPOSE:  THIS PROGRAM PRODUCES A QUARTERLY SUBJECT INDEX              */
/*            PRINTOUT FOR BOTH AIM AND ARM.  THE MASTER DATA SET          */
/*            THAT HAS BEEN GENERATED FOR THE QUARTER IS THE MAIN          */
/*            INPUT FOR THE PROGRAM.  THE NECESSARY DATA IS TAKEN          */
/*            FROM MASTER AND PUT ON A TEMPORARY DISK FILE.  WE THEN       */
/*            CALL FOR THE SORT/MERGE, WHICH SORTS THE RECORDS ON          */
/*            THE DISK AND PUTS THE SORTED RECORDS ON ANOTHER DISK         */
/*            FILE.  THE SORTED DATA IS PRINTED UNDER A FORMAT WHICH       */
/*            THE VOCATIONAL CENTER SPECIFIES AND THIS IS THE INDEX        */
/*            PRINTOUT FOR THE PUBLICATION.  NOW, EACH RECORD ON THE       */
/*            SORTED DISK FILE IS SHORTENED AND SAVED ON MAGNETIC          */
/*            TAPE FOR THE ANNUAL INDEX.                                   */
/*                                                                         */
/*  DEFINITION OF VARIABLES:                                               */
/*     1. BT -- CHARACTER DENOTING THE BEGINNING OF A SHORT TITLE          */
/*              SEGMENT (<).                                               */
/*     2. COL -- THE NUMBER OF COLUMNS OF OUTPUT IN THE INDEX              */
/*               PRINTOUT.                                                 */
/*     3. DESC -- DESCRIPTOR STRING FROM THE MASTER TAPE.                  */
/*     4. DUMMY -- STRING USED FOR INPUTING AND OUTPUTING TAPE AND         */
/*                 DISK RECORDS.                                           */
/*     5. ED_NO -- ED OR ACCESSION NUMBER.                                 */
/*     6. ET -- CHARACTER DENOTING THE END OF A SHORT TITLE SEGMENT.       */
/*              (>)                                                        */
/*     7. HEAD -- A STRING WHICH HEADS EVERY PAGE OF THE INDEX. IT         */
/*                GIVES AIM OR ARM, THE SEASON, AND THE YEAR.  (FOR        */
/*                EXAMPLE, AIM/FALL 69)                                    */
/*     8. IDENT -- IDENTIFIER STRING FROM THE MASTER TAPE.                 */
/*     9. IHESRTA -- THE PL/I CALL TO SORT/MERGE.                          */
/*    10. LENTH -- AN ARRAY USED FOR STORING LENGTHS OF STRINGS.           */
/*    11. LH -- LENGTH OF -HEAD- (NO. 7).                                  */
/*    12. LINEH1 -- THE LINE NUMBER IN THE INDEX PRINTOUT ON WHICH         */
/*                  -HEAD- AND THE PAGE NUMBER APPEAR ON THE TITLE         */
/*                  PAGE.                                                  */
/*    13. LINEH2 -- SAME AS LINEH1 EXCEPT IT'S FOR ALL SUCCEEDING          */
/*                  PAGES OF THE INDEX                                     */
/*    14. LINEP1 -- THE LINE NUMBER IN THE INDEX PRINTOUT ON WHICH         */
/*                  THE WORD 'PAGE' APPEARS ABOVE EVERY COLUMN.  THIS*/
/*                  IS FOR THE TITLE PAGE ONLY.                            */
/*    15. LINEP2 -- SAME AS LINEP1 EXCEPT IT'S FOR ALL SUCCEEDING          */
/*                  PAGES OF THE INDEX.                                    */
/*    16. LINET -- THE LINE NUMBER IN THE INDEX PRINOUT ON WHICH THE */
/*                 TITLE APPEARS.  WE DO NOT PUT A TITLE ON THE TITLE*/
/*                 PAGE.                                                   */
/*    17. LTH -- LENGTH OF THE TITLE OF THE INDEX.                         */
/*    18. L_DESC -- LENGTH OF THE DESCRIPTOR STRING.                       */
/*    19. L_IDENT -- LENGTH OF THE IDENTIFIER STRING.                      */
/*    20. L_TITLE -- LENGTH OF THE TITLE OF THE DOCUMENT.                  */
/*    21. MARG -- LEFT-HAND MARGIN OF THE PAGE.                            */
/*    22. MASTER -- NAME OF THE MASTER TAPE.                               */
/*    23. MG -- ARRAY OF MARGINS FOR THE COLUMNS IN THE INDEX              */
/*              PRINTOUT.                                                  */
/*    24. OUT -- ARRAY OF STARRED IDENTIFIERS AND DESCRIPTORS FOR A        */
/*               DOCUMENT.                                                 */
/*    25. PAGE -- PAGE NUMBER OF THE INDEX PRINTOUT.                       */
/*    26. PAGE_NO -- RESUME PAGE FROM MASTER.                              */
/*    27. PUB_DATE -- PUBLICATION DATE FROM MASTER.
/*    28. QUE -- TELLS US WHETHER THE INDEX TITLE PAGE IS ON THE      */00000630
/*               LEFT OR RIGHT SIDE OF THE PUBLICATION.                */00000640
/*    29. RETCDE -- CODE RETURNED BY THE SORT PROGRAM.  '0' MEANS      */00000650
/*                  SORT WENT WELL; '16' MEANS SORT HAD TROUBLE.       */00000660
/*    30. RFIELD -- DESCRIBES THE RECORDS BEING SORTED FOR THE SORT    */00000670
/*                  PROGRAM.                                           */00000680
/*    31. SFIELD -- DESCRIBES THE PART(S) OF THE RECORD TO BE SORTED   */00000690
/*                  FOR THE SORT PROGRAM.                              */00000700
/*    32. SIZE -- THE NUMBER OF LINES THAT THE INDEX PRINTOUT MAY      */00000710
/*                HAVE PER PAGE.  IT COUNTS FROM LINE 1 AND STOPS AT   */00000720
/*                THE LAST ALLOWABLE LINE ON WHICH INDEX PRINTOUT IS   */00000730
/*                ALLOWED.                                             */00000740
/*    33. SORTIN -- DEFINES THE DISK SUBSET FILE WITH THE UNSORTED     */00000750
/*                  RECORDS.                                           */00000760
/*    34. SORTOUT -- DEFINES THE DISK SUBSET FILE WITH THE SORTED      */00000770
/*                   RECORDS.                                          */00000780
/*    35. SPACE -- THE NUMBER OF SPACES BETWEEN THE VT AND ED NUMBER   */00000790
/*                 IN THE INDEX PRINTOUT.                              */00000800
/*    36. SUM1 -- THE NUMBER OF BYTES ON A MASTER RECORD BETWEEN THE   */00000810
/*                RESUME PAGE AND THE TITLE.  THIS IS REALLY THE       */00000820
/*                SUM OF THE NAMES OF THE PERSONAL AUTHORS IF THERE    */00000830
/*                ARE ANY.                                             */00000840
/*    37. SUM2 -- THE NUMBER OF BYTES ON A MASTER RECORD BETWEEN THE   */00000850
/*                TITLE AND THE PUBLICATION DATE.                      */00000860
/*    38. SUM3 -- THE NUMBER OF BYTES ON A MASTER RECORD BETWEEN THE   */00000870
/*                PUBLICATION DATE AND THE DESCRIPTORS.  THIS IS       */00000880
/*                REALLY THE SUM OF THE LENGTH  OF THE DESCRIPTIVE     */00000890
/*                NOTE AND THE LENGTH OF THE PAGE VARIABLE.            */00000900
/*    39. SZ -- THE RIGHT-HAND MARGIN OF THE INDEX PRINTOUT PAGES.     */00000910
/*    40. TITL -- THE TITLE OF THE INDEX, 'SUBJECT AND IDENTIFIER      */00000920
/*                INDEX'.                                              */00000930
/*    41. TITLE -- TITLE OF A DOCUMENT FROM THE MASTER RECORD.         */00000940
/*    42. TLE -- ARRAY OF SHORT TITLE SEGMENTS.                        */00000950
/*    43. VT_NO -- DOCUMENT NUMBER FROM MASTER TAPE.                   */00000960
/*    44. WIDTH -- WIDTH OF THE COLUMNS IN THE INDEX PRINTOUT.         */00000970
/*    45. BLK -- NUMBER OF PIECES OF WIDTH -WIDTH- THAT A KEYWORD      */00000980
/*               IS BROKEN UP INTO TO FIT IN A COLUMN IN THE INDEX     */00000990
/*               PRINTOUT.                                             */00001000
/*    46. BLKT -- NUMBER OF PIECES OF WIDTH -WIDTH-2- THAT A TITLE     */00001010
/*                OF A DOCUMENT IS BROKEN UP INTO TO FIT IN A COLUMN   */00001020
/*                IN THE INDEX PRINTOUT.                               */00001030
/*    47. COL# -- IS A TWO DIMENSIONAL ARRAY THAT REPRESENTS AN        */00001040
/*                INDEX PRINTOUT PAGE.  ITS FIRST DIMENSION GIVES THE*/00001050
/*                COLUMN NUMBER OF SOME COLUMN ON A PAGE, AND ITS      */00001060
/*                SECOND DIMENSION IS A LINE NUMBER IN THE COLUMN      */00001070
/*                REPRESENTED BY THE FIRST DIMENSION.  I PUT A PAGE    */00001080
/*                OF THE INDEX INTO THIS ARRAY AND THEN PRINT THE      */00001090
/*                ARRAY.  THIS IS THE ONLY WAY TO PRINT A WHOLE PAGE   */00001100
/*                AT ONCE WHEN DATA IS ASSIGNED TO A PAGE IN VERTICAL*/00001110
/*                COLUMNS AND THE COMPUTER PRINTER PRINTS              */00001120
/*                HORIZONTALLY ONLY.                                   */00001130
/*    48. IMAX -- IS THE NUMBER OF COLUMNS PLUS ONE.                   */00001140
/*    49. IMP -- INDICATES WHETHER A NEW KEYWORD IS BEING STARTED      */00001150
/*               IN THE INDEX PRINTOUT.                                */00001160
/*    50. IND -- INDICATES WHAT   COLUMN OF A PAGE IS BEING FILLED.    */00001170
/*    51. KNT -- IS A COUNT OF THE NUMBER OF LINES IN A BLOCK OF DATA*/00001180
/*               BY BLOCK I MEAN, VT NUMBER, ED NUMBER  AND RESUME     */00001190
/*               PAGE ON ONE LINE AND TITLE BROKEN UP INTO -BLKT-      */00001200
/*               LINES.  ALSO, IF WE ARE LISTING THE FIRST DOCUMENT    */00001210
/*               FOR A KEYWORD, WE COUNT THE KEYWORD BROKEN UP INTO    */00001220
/*               -BLK- LINES, PLUS ONE FOR THE BLANK LINE BETWEEN      */00001230
/*               KEYWORD LISTINGS.                                     */00001240
/*    52. LN -- IS THE LAST LINE IN A PARTICULAR COLUMN THAT HAS      */00001250
/*              BEEN FILLED WITH INDEX PRINTOUT.                      */00001260
/*    53. NN -- IS A SUBSCRIPT USED FOR THE SECOND DIMENSION OF       */00001270
/*              -COL#-                                                */00001280
/*    54. OUTS -- NAME OF OUTPUT FILE WHERE THE INDEX IS PRINTED.     */00001290
/*    55. PC -- THE PIECES OF WIDTH -WIDTH- INTO WHICH A KEYWORD IS   */00001300
/*              BROKEN UP.                                            */00001310
/*    56. PREV -- KEEPS THE NAME OF THE CURRENT KEYWORD BEING LISTED  */00001320
/*                IN THE INDEX.                                       */00001330
/*    57. PUB -- THE YEAR SECTION OF THE PUBLICATION DATE.            */00001340
/*    58. SKPH -- GIVES THE MARGIN FOR VARIABLE -HEAD- SO I KNOW      */00001350
/*                WHERE TO POSITION IT.                               */00001360
/*    59. SKPP -- GIVES THE MARGIN FOR THE PAGE NUMBER ON THE INDEX   */00001370
/*                PRINTOUT.                                           */00001380
/*    60. SUB -- NAME OF THE MAGNETIC TAPE DATA SET ON WHICH THE      */00001390
/*               SHORTENED RECORDS FOR THE ANNUAL INDEX ARE SAVED.    */00001400
/*    61. TLP -- THE PIECES OF WIDTH -WIDTH-2- INTO WHICH A TITLE IS  */00001410
/*               BROKEN UP.                                           */00001420
/*                                                                    */00001430
/*  INPUT TO THE PROGRAM:                                             */00001440
/*    -MASTER-, THE MASTER TAPE DATA SET                              */00001450
/*      THE MASTER TAPE DATA SET GENERATED FOR THE QUARTER IS ONE     */00001460
/*      INPUT TO THE PROGRAM.  IT IS REFERENCED BY DD CARD IN THE     */00001470
/*      JCL.                                                          */00001480
/*    -SYSIN-, INPUT ON CARDS                                         */00001490
/*      WE NEED TO INPUT TO THE PROGRAM THE OUTPUT SPECIFICATIONS     */00001500
/*      FOR THE INDEX PRINTOUT.  THE FOLLOWING LIST TELLS WHAT DATA   */00001510
/*      IS READ IN.  THE DATA IS ONE CONTINUOUS STREAM AND TAKES AS   */00001520
/*      MANY CARDS AS NECESSARY.                                      */00001530
/*      FIELD 1 IS THE PAGE NUMBER FOR THE FIRST PAGE OF THE INDEX    */00001540
/*              PRINTOUT.  F(4,0) FORMAT.                             */00001550
/*      FIELD 2 IS THE SIZE OF THE PRINTOUT PAGE.  F(4,0) FORMAT.     */00001560
/*      FIELD 3 IS THE COLUMN WIDTH.  F(4,0) FORMAT.                  */00001570
/*      FIELD 4 IS THE LEFT-HAND MARGIN OF THE PAGE. F(4,0) FORMAT.   */00001580
/*      FIELD 5 IS THE RIGHT-HAND MARGIN OF THE PAGE. F(4,0)          */00001590
/*      FIELD 6 IS THE LINE ON THE TITLE PAGE WHERE THE PAGE NUMBER   */00001600
/*              AND HEADING BELONG. F(4,0)                            */00001610
/*      FIELD 7 IS THE LINE ON THE TITLE PAGE WHERE THE WORD 'PAGE'   */00001620
/*              IS WRITTEN ABOVE EACH COLUMN. F(4,0)                  */00001630
/*      FIELD 8 IS THE LINE ON ALL SUCCEEDING PAGES WHERE THE PAGE    */00001640
/*              NUMBER AND HEADING BELONG. F(4,0)                     */00001650
/*      FIELD 9 IS THE LINE ON ALL SUCCEEDING PAGES WHERE THE TITLE   */00001660
/*              SHOULD APPEAR.  F(4,0)                                */00001670
/*      FIELD 10 IS THE LINE ON ALL SUCCEEDING PAGES WHERE THE WORD   */00001680
/*              'PAGE' IS WRITTEN ABOVE EACH COLUMN.  F(4,0)          */00001690
/*      FIELD 11 IS SPACE BETWEEN THE VT AND ED NUMBER IN THE         */00001700
/*              PRINTOUT.  F(4,0)                                     */00001710
/*      FIELD 12 IS THE NUMBER OF COLUMNS IN THE INDEX PRINTOUT.      */00001720
/*              F(4,0)                                                */00001730
/*      FIELD 13 IS THE MARGIN FOR EACH COLUMN MEASURED FROM COLUMN   */00001740
/*              ONE.  FORMAT IS THE NUMBER OF COLUMNS TIMES F(4,0).   */00001750
/*      FIELD 14 IS THE LENGTH OF THE INDEX TITLE.  F(4,0)            */00001760
/*      FIELD 15 IS THE INDEX TITLE. FORMAT IS A(FIELD 14'S NO.).     */00001770
/*      FIELD 16 IS THE LENGTH OF THE HEADING.  F(4,0)                */00001780
/*      FIELD 17 IS THE HEADING (EX: FALL/ARM 69).  FORMAT IS         */00001790
/*              A(FIELD 16'S NO.).                                    */00001800
/*      FIELD 18 IS AN 'L' OR 'R'. 'L' SAYS THAT THE FIRST PAGE OF    */00001810
/*              THE INDEX WILL BE ON THE LEFT SIDE OF THE PUBLICATION. */00001820
/*              'R' INDICATES THE RIGHT SIDE.  FORMAT A(1).           */00001830
/*                                                                    */00001840
/*  SPECIAL NOTE:                                                     */00001850
/*    THE OUTPUT FILE SHOULD BE THE PRINTER UNTIL THE VOCATIONAL      */00001860
/*   CENTER SEES AND APPROVES THE INDEX PRINTOUT.  WE DEFINE IT AS  */00001870
/*   '//OUTS DD SYSOUT=A,DCB=(RECFM=F,LRECL=133,BLKSIZE=133)'. THEN */00001880
/*   OUTS SHOULD BE DEFINED AS A TAPE.  THE OUTPUT TAPE IS THEN     */00001890
/*   LISTED ON THE 360/50 FOR AN UPPER AND LOWER CASE PRINTOUT. THE */00001900
/*   JCL FOR THIS 50 RUN IS IN THE MISCELLANEOUS PROGRAM SECTION.   */00001910
/*                                                                  */00001920
/*******************************************************************/ 00001930
                                                                        00001940
                                                                        00001950
SUBIND: PROCEDURE OPTIONS (MAIN);                                       00001960
    DECLARE                                                             00001970
      BT                  CHARACTER (1) INITIAL ('<'),                  00001980
      COL                                                               00001990
          FIXED BINARY (31,0),                                          00002000
      DESC                                                              00002010
          CHARACTER (660) VARYING,                                      00002020
      DISECT                                                            00002030
          ENTRY,                                                        00002040
      DUMMY               CHARACTER (3736) VARYING,                     00002050
      ED_NO               CHARACTER (20),                               00002060
      ET                  CHARACTER (1) INITIAL ('>'),                  00002070
      HEAD                CHARACTER (13) VARYING,                       00002080
      I                                                                 00002090
          FIXED BINARY (31,0),                                          00002100
      IDENT               CHARACTER (396) VARYING,                      00002110
      IHESRTA                                                           00002120
          ENTRY(CHARACTER(48),CHARACTER(34),FIXED BINARY(31,0),         00002130
               FIXED BINARY(15,0)),                                     00002140
      J                                                                 00002150
          FIXED BINARY (31,0),                                          00002160
      L                                                                 00002170
          FIXED BINARY (31,0),                                          00002180
      LENTH(10)           FIXED BINARY (31,0),                          00002190
      LH                  FIXED BINARY (31,0),                          00002200
      LINEH1              FIXED BINARY (31,0),                          00002210
      LINEH2              FIXED BINARY (31,0),                          00002220
      LINEP1              FIXED BINARY (31,0),                          00002230
      LINEP2              FIXED BINARY (31,0),                          00002240
      LINET               FIXED BINARY (31,0),                          00002250
      LTH                                                               00002260
          FIXED BINARY (31,0),                                          00002270
      L_DESC              FIXED BINARY (31,0),                          00002280
      L_IDENT             FIXED BINARY (31,0),                          00002290
      L_TITLE             FIXED BINARY (31,0),                          00002300
      M                                                                 00002310
          FIXED BINARY (31,0),                                          00002320
      MARG                FIXED BINARY (31,0),                          00002330
      MASTER                                                            00002340
          FILE SEQUENTIAL UNBUFFERED INPUT RECORD,                      00002350
      MC(3)               FIXED BINARY (31,0),                          00002360
      N                                                                 00002370
          FIXED BINARY (31,0),                                          00002380
      NUMB                FIXED BINARY (31,0),                          00002390
      OUT(15) CHAR(200) VAR,                                            00002400
      PAGE                                                              00002410
          FIXED BINARY (31,0),                                          00002420
      PAGE_NO             CHARACTER (4),                                00002430
      PUB_DATE                                                          00002440
          CHARACTER (7),                                                00002450
      QUE                 CHARACTER (1),                                00002460
      RETCDE                                                            00002470
          FIXED BINARY (15,0),                                          00002480
```

```
         RFIELD
             CHARACTER (34),
         SFIELD
             CHARACTER (48),
         SIZE
             FIXED BINARY (31,0),
         SORTIN
             FILE SEQUENTIAL RECORD,
         SORTOUT
             FILE SEQUENTIAL RECORD,
         SPACE              FIXED BINARY (31,0),
         SUM1
             FIXED BINARY (31,0),
         SUM2
             FIXED BINARY (31,0),
         SUM3
             FIXED BINARY (31,0),
         SZ                 FIXED BINARY (31,0),
         TITL
             CHARACTER (50) VARYING,
         TITLE              CHARACTER (396) VARYING,
         TLE(3)
             CHARACTER (396) VARYING,
         VT_NO
             CHARACTER (10) VARYING,
         WIDTH
             FIXED BINARY (31,0),
       ;

       OPEN FILE (MASTER),
            FILE (ERRE) OUTPUT PRINT STREAM LINESIZE (125),
            FILE (SORTIN) OUTPUT:
       ON ENDFILE (MASTER) GO TO SORT;


   /*  READ A RECORD FROM THE MASTER TAPE.  SEARCH FOR THE DIVIDER   (||)
       IN THE RECORD.                                                   */


 READ_MASTER:
       READ FILE (MASTER) INTO (DUMMY);
       N = INDEX(DUMMY,'||');
       IF N = 0
         THEN DO:
              VT_NO = SUBSTR(DUMMY,1,10);
              PUT FILE (ERRE) EDIT
                ('MISSING DIVIDER IN DOCUMENT ',VT_NO)
                (SKIP(2),A(28),A(10));
              GO TO READ_MASTER;
              END;


   /*  FIND THE NUMBER OF POSITIONS FROM THE RESUME PAGE TO THE TITLE,
       SO THAT THE TITLE CAN BE PULLED OFF OF MASTER.  -SUM1- IS THIS
       VALUE.  ALSO, GET THE TITLE LENGTH.                              */


       SUM1 = 0;
       GET STRING (DUMMY) EDIT (NUMB,(LENTH(I) DO I = 1 TO NUMB),
         L_TITLE)  (X(N+1),F(4),(NUMB+1) F(4));
       N = N + 9 + (NUMB * 4);
       IF NUMB¬= 0 THEN DO;
```

```
                        LOOP1: DO I = 1 TO NUMB;
                               SUM1 = SUM1 + LENTH(I);
                               END LOOP1;
                        END;


 /*  FIND THE NUMBER OF POSITIONS FROM THE TITLE TO THE PUBLICATION
     DATE SO THAT THE DATE CAN BE PULLED OFF OF MASTER.  -SUM2- IS
     THIS VALUE.                                                       */


        SUM2 = 0;
LOOP2:  DO J = 1 TO 2;
        GET STRING (DUMMY) EDIT (NUMB,(LENTH(I) DO I=1 TO NUMB))
          (X(N),F(4),(NUMB) F(4));
        N = N + 4 + (4 * NUMB);
        IF NUMB ¬= 0 THEN DO;
                          INNER: DO L = 1 TO NUMB;
                                 SUM2 = SUM2 + LENTH(L);
                                 END INNER;
                          END;
        END LOOP2;
LOOP3:  DO I = 1 TO 8;
        GET STRING (DUMMY) EDIT (NUMB)
          (X(N),F(4));
        SUM2 = SUM2 + NUMB;
        N = N + 4;
        END LOOP3;
        N = N + 4;


 /*  FIND THE NUMBER OF POSITIONS FROM THE DATE TO THE IDENTIFIERS
     AND DESCRIPTORS SO THAT THESE STRINGS CAN BE PULLED OFF OF MASTER.
     -SUM3- IS THIS VALUE.  ALSO, GET THESE STRINGS' LENGTHS.          */


        SUM3 = 0;
LOOP3A: DO I = 1 TO 2;
        GET STRING (DUMMY) EDIT (NUMB)  (X(N),F(4));
        SUM3 = SUM3 + NUMB;
        N = N + 4;
        END LOOP3A;
        GET STRING (DUMMY) EDIT (L_DESC,L_IDENT) (X(N),2 F(4));


 /*  WITH ALL THIS FROM MASTER, THE NECESSARY DATA CAN BE RETRIEVED
     FROM THE TAPE.                                                    */


        GET STRING (DUMMY) EDIT (VT_NO,ED_NO,PAGE_NO,TITLE,PUB_DATE,
                                DESC,IDENT)
           (A(10),X(17),A(20),X(7),A(4),X(SUM1),A(L_TITLE),X(SUM2),
           A(7),X(SUM3),A(L_DESC),A(L_IDENT));
        PUT SKIP (4);
        PUT EDIT (VT_NO,ED_NO,PAGE_NO,TITLE,PUB_DATE,DESC,IDENT) (A);


 /*  FORM THE SHORT TITLE FROM THE TITLE PULLED OFF OF MASTER.         */


        I = 0;
RETRN:  N = INDEX(TITLE,BT);
```

```
         M =  INDEX(TITLE,ET);
         IF (N=0) & (M=0) THEN GO TO LOOP4;
         I = I + 1;
         TLE(I) = SUBSTR(TITLE,N+1,M-N-1);
         IF SUBSTR(TITLE,M+1,1) ¬= ' ' THEN TLE(I) = TLE(I) ||
                                              SUBSTR(TITLE,M+1,1);
         TITLE = SUBSTR(TITLE,M+2);
         GO TO RETRN;
LOOP4:   DO J = 1 TO I;
         IF J = 1 THEN TITLE = TLE(1);
                  ELSE TITLE = TITLE || ' ' || TLE(J);
         END LOOP4:
         PUT DATA (TITLE);


  /*  BREAK UP THE DESCRIPTOR AND IDENTIFIER STRINGS INTO THE INDIVIDUAL
      KEYWORDS.  STORE THE STARRED ONES IN -OUT-.                      */


WHOLE:   I = 0;
         CALL DISECT(DESC,I,OUT);
         IF L_IDENT = 0 THEN GO TO LOOP5;
         DESC = IDENT;
         CALL DISECT(DESC,I,OUT);


  /*  PUT THE STARRED KEYWORDS WITH THEIR CORRESPONDING INFORMATION
      ON THE DISK SUBSET FILE IN INDIVIDUAL RECORDS PER KEYWORD.       */


LOOP5:   DO J = 1 TO I:
NONE:    PUT STRING (DUMMY) EDIT (OUT(J),VT_NO,ED_NO,PAGE_NO,PUB_DATE,
                                      TITLE)
           (A(200),A(10),A(20),A(4),A(7),A):
         WRITE FILE (SORTIN) FROM (DUMMY);
         IDENT = OUT(J);
         PUT DATA (IDENT):
FL5:     END LOOP5:

         GO TO READ_MASTER;


  /*  THIS PROCEDURE BREAKS UP THE DESCRIPTOR  AND IDENTIFIER STRINGS
      INTO THE INDIVIDUAL KEYWORDS AND STORES THE STARRED ONES IN -OUT-.
      IT ALSO CHECKS THAT EACH KEYWORD FITS WITHIN THE 132 BYTE LENGTH
      MAXIMUM THAT I HAVE CHOSEN.                                      */


DISECT: PROCEDURE (STRING,I,OUT);
         DECLARE
           STRING                CHARACTER (660) VARYING,
           I                     FIXED BINARY (31,0),
           OUT(15) CHAR(200) VAR,
           REP                   CHARACTER (1) INITIAL ('"'),
         ;
SRC:     N = INDEX(STRING,';');
         IF N = 0 THEN DO;
                           N = LENGTH(STRING);
                           N = N + 1;
                           END;
           IF N>200
           THEN DO;
```

```
               PUT FILE (ERRE) EDIT
                ('KEYWORD TOO LARGE IN DOCUMENT ',VT_NO)
                (SKIP(2),A(30),A(10));
               GO TO TRN;
               END:
        IF SUBSTR(STRING,1,1)='*' THEN DO;
                                        I = I + 1;
                                        OUT(I)=SUBSTR(STRING,2,N-2)||
                                              REP:
                                        END;
TRN:    N = INDEX(STRING,';');
        IF N = 0 THEN RETURN;
        STRING = SUBSTR(STRING,N+2);
        GO TO SRC:
        END DISECT;


 /*  MAKE THE PROPER PREPARATIONS AND CALL SORT.  PRINT THE SORT RETURN
     CODE: '0' INDICATES A GOOD SORT.                                 */


SORT:   CLOSE FILE (SORTIN),
              FILE (MASTER);
        SFIELD=' SORT FIELDS=(5,200,CH,A,208,7,CH,A),SIZE=E9999 ';
        RFIELD=' RECORD TYPE=V,LENGTH=(641,,,245) ';
        CALL IHESRTA(SFIELD,RFIELD,45056,RETCDE);
        PUT FILE (SYSPRINT) EDIT ('RETURN CODE = ',RETCDE)
          (SKIP(3),A(14),F(3));


 /*  READ IN THE PAGE SPECIFICATIONS FOR THE QUARTERLY INDEX PRINTOUT.
                                                                      */


        GET FILE (SYSIN) EDIT (PAGE,SIZE,WIDTH,MARG,SZ,LINEH1,LINEP1,
         LINEH2,LINEFT,LINEP2,SPACE,COL,(MG(N) DO N = 1 TO COL),LTH,
         TITL,LH,HEAD,QUE)
          (13 F(4),(COL) F(4),A(LTH),F(4),A(LH),A(1));


 /*  THE PRINTING OF THE INDEX AND THE SHORTENING OF THE SUBSET FILE
     ARE ACCOMPLISHED IN THIS BLOCK.                                  */


WORK:   BEGIN;
        DECLARE
          BLK
              FIXED BINARY (31,0),
          BLKT
              FIXED BINARY (31,0),
          BREAK            ENTRY(,FIXED BINARY (31,0),,),
          COL#(COL,SIZE-LINEP2-1)  CHARACTER (WIDTH+3),
          IMAX
              FIXED BINARY (31,0),
          IMP              FIXED BINARY (31,0),
          IND
              FIXED BINARY (31,0),
          K
              FIXED BINARY (31,0),
          KK               FIXED BINARY (31,0),
          KNT              FIXED BINARY (31,0),
          LL               FIXED BINARY (31,0),
        LN
             FIXED BINARY (31,0),
        NN                    FIXED BINARY (31,0),
        OUTS
             FILE,
        PC(14)                CHARACTER (WIDTH) VARYING,
        PREV CHAR(200) VAR INITIAL('99'),
        PUB
             CHARACTER (2),
        SKPH                  FIXED BINARY (31,0),
        SKPP                  FIXED BINARY (31,0),
        SUB
             FILE SEQUENTIAL RECORD,
        TLP(14)               CHARACTER (WIDTH) VARYING,
      ;


 /*  INITIALIZE VARIABLES AS NEEDED FOR PRINTING.  OPEN THE OUTPUT
     FILE WHICH RECEIVES THE INDEX PRINTOUT.                          */


        ON ENDFILE (SORTOUT) GO TO SHORTEN;
        OPEN FILE (SORTOUT),
             FILE (OUTS) OUTPUT STREAM PRINT PAGESIZE(SIZE+4)
        ;
BLKOT:  DO I = 1 TO SIZE-LINEP2-1;
        DO J = 1 TO COL;
        COL#(J,I) = ' ';
        END BLKOT;
        K = PAGE;
        IND =1;
        IMAX = COL + 1;
        LN = LINEP1;
        NN = 0;
        IF QUE = 'L' THEN DO;
                             SKPP = MARG;
                             SKPH = SZ - LH + 1;
                             END;
                       ELSE DO;
                             SKPH = MARG;
                             SKPP = SZ - 3;
                             END;


 /*  PRINT THE HEADING FOR THE TITLE PAGE.                              */


        PUT FILE (OUTS) EDIT (PAGE,HEAD)
          (PAGE,LINE(LINEH1),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
          A(LH));
        PUT FILE (OUTS) SKIP (LINEP1-LINEH1);
        LP8: DO I = 1 TO COL;
             PUT FILE (OUTS) EDIT ('PAGE')
               (SKIP(0),COLUMN(MG(I)+WIDTH-1),A(4));
             END LP8;
        PUT FILE (OUTS) SKIP(1);


 /*  READ IN A RECORD FROM THE SORTED DISK SUBSET FILE.  BREAK UP THE
     RECORD INTO THE INDIVIDUAL INFORMATION.  SEARCH FOR THE MARK IN
     THE KEYWORD THAT INDICATES WHERE THE BLANKS ON THE RIGHT BEGIN AND
     CUT THE KEYWORD AT THIS POINT.  BREAK UP THE TITLE INTO STRINGS
```

```
      THAT FIT WITHIN THE COLUMN WIDTH SPECIFIED.                      */


 READ:   READ FILE (SORTOUT) INTO (DUMMY);
         L = LENGTH(DUMMY);
         GET STRING (DUMMY) EDIT (IDENT,VT_NO,ED_NO,PAGE_NO,PUB_DATE,
               TITLE)(A(200),A(10),A(20),A(4),A(7),A(L-241));
         N = INDEX(IDENT,'"');
         IDENT = SUBSTR(IDENT,1,N-1);
         CALL BREAK (TITLE,WIDTH-2,TLP,BLKT);


  /*  IF THE RECORD IS THE START OF DATA FOR A NEW KEYWORD, THEN BREAK
      UP THE KEYWORD INTO STRINGS OF LENGTH THAT MATCH THE COLUMN WIDTH.
      INDICATE A NEW KEYWORD'S START WITH -IMP-.  INITIALIZE -KNT-,THE
      NUMBER OF LINES IN THIS BLOCK OF DATA.                           */


         IF IDENT ¬= PREV
           THEN DO;
                PREV = IDENT;
                IF (N-1) > WIDTH THEN CALL BREAK(IDENT,WIDTH,PC,BLK);
                                 ELSE DO;
                                      BLK = 1;
                                      PC(1) = IDENT;
                                      END;
                KNT = BLK + 2 + BLKT;
                IMP = 1;
                END;


  /*  IF THE SAME KEYWORD IS CONTINUING,SET VALUES IN -IMP- AND -KNT-
      FOR THIS SITUATION.                                              */


           ELSE DO;
                KNT = BLKT + 1;
                IMP = 0;
                END;


  /*  IF THE BLOCK OF DATA CANNOT FIT INTO THE CURRENT COLUMN BEING
      FILLED WITHOUT EXCEDING THE PAGE LENGTH, A NEW COLUMN MUST BE
      STARTED.                                                         */


         IF (LN + KNT) > SIZE THEN GO TO HEADR;


  /*  IF THIS IS A NEW KEYWORD, PRINT ITS NAME IN THE INDEX.           */


 RETURN: IF IMP = 1 THEN DO;
                         NN = NN + 1;
                         LP6: DO I = 1 TO BLK;
                              COL#(IND,NN) = PC(I);
                              NN = NN + 1;
                              END LP6;
                         END;


  /*  PRINT THE DOCUMENT NUMBER AND OTHER DATA FOR THE DOCUMENT BEING
```

```
     LISTED UNDER THE KEYWORD.                                          */


        PUT STRING (COL#(IND,NN)) EDIT (VT_NO,ED_NO,PAGE_NO)
         (A(10),X(SPACE),A(20),X(WIDTH-30-SPACE-1),A(4));
        NN = NN + 1;
LP7:    DO I = 1 TO BLKT;
        COL#(IND,NN) = '  ' || TLP(I);
        NN = NN + 1;
        END LP7;
        LN = LN + KNT;
        GO TO READ;


 /*  A BRANCH HERE STARTS A NEW COLUMN BECAUSE THE CURRENT COLUMN IS AS
     FULL AS CAN BE.  CHANGE -IND- TO THE NEXT COLUMN NUMBER ON THE
     PAGE.                                                              */


HEADR:  IND = IND + 1;


 /*  IF THE LAST COLUMN ON THE PAGE IS FULL, START A NEW PAGE AS WELL
     AS A NEW COLUMN.
     SET THE COLUMN NUMBER TO ONE.  PRINT OUT THE PAGE JUST COMPLETED
     ON THE OUTPUT FILE.  INCREMENT THE PAGE NUMBER OF THE INDEX AND
     PRINT A TITLE ON THE NEW PAGE.                                     */


        IF IND = IMAX
          THEN DO;
                IND = 1;
                HD1: DO LL = 1 TO SIZE-LINEP2-1;
                     PUT FILE (OUTS) SKIP(1);
                     LP9: DO KK = 1 TO COL;
                          PUT FILE (OUTS) EDIT (COL#(KK,LL))
                           (SKIP(0),COLUMN(MG(KK)),A(WIDTH+3));
                          COL#(KK,LL) = ' ';
                          END LP9;
                     END HD1;
                PAGE = PAGE + 1;
                IF SKPP =MARG THEN DO;
                                   SKPP = SZ - 3;
                                   SKPH = MARG;
                                   END;
                              ELSE DO;
                                   SKPP = MARG;
                                   SKPH = SZ - LH + 1;
                                   END;
                PUT FILE (OUTS) EDIT (PAGE,HEAD,TITL)
                 (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(
                  SKPH),A(LH),LINE(LINET),X((SZ-LTH-MARG+1)/2+MARG-1),
                  A(LTH));
                PUT FILE (OUTS) SKIP(LINEP2-LINET);
                LP10: DO LL = 1 TO COL;
                      PUT FILE (OUTS) EDIT ('PAGE')
                        (SKIP(0),COLUMN(MG(LL)+WIDTH-1),A(4));
                      END LP10;
                PUT FILE (OUTS) SKIP(1);
                END;


 /*  NEW PAGE OR NOT, INITIALIZE THE LINE COUNTERS FOR A NEW COLUMN. */


         IF IMP = 1 THEN DO;
                         NN = 0;
                         LN = LINEP2;
                         END;
                    ELSE DO;
                         NN = 1;
                         LN = LINEP2 + 1;
                         END;
         IF K = PAGE THEN LN = LN + LINEP1 - LINEP2;
         GO TO RETURN;


 /*  WHEN THE SORTED FILE HAS BEEN READ COMPLETELY, PRINT OUT THE PAGE
     BEING FILLED ON THE OUTPUT FILE.  INDICATE THE LAST PAGE NUMBER OF
     INDEX, AND CLOSE THE OUTPUT FILE.                                 */


SHORTEN:
         DO I = 1 TO SIZE-LINEP2-1;
         PUT FILE (OUTS) SKIP(1);
         LP11: DO J = 1 TO COL;
                 PUT FILE (OUTS) EDIT (COL#(J,I))
                   (SKIP(0),COLUMN(MG(J)),A(WIDTH+3));
                 END LP11;
         END SHORTEN;
         PUT FILE (OUTS) PAGE;
         PUT FILE (SYSPRINT) EDIT ('LAST PAGE USED IS ',PAGE)
           (SKIP(3),A(18),F(4));
         CLOSE FILE (OUTS),
                 FILE (SORTOUT);


  /*  SHORTEN THE RECORDS ON SORTED SUBSET FILE AND FORM A NEW SUBSET
      FILE ON TAPE.                                                    */


         ON ENDFILE (SORTOUT) GO TO STOP;
         OPEN FILE (SORTOUT),
                FILE (SUB) OUTPUT;
         PUT FILE (SYSPRINT) PAGE;
  RD:    READ FILE (SORTOUT) INTO (DUMMY);
         GET STRING (DUMMY) EDIT (IDENT,VT_NO,PUB)
                (A(200),A(10),X(29),A(2));
         VT_NO = SUBSTR(VT_NO,4,3) || SUBSTR(VT_NO,8,3);
         PUT STRING (DUMMY) EDIT (IDENT,PUB,VT_NO)
                (A(200),A(2),A(6));
         WRITE FILE (SUB) FROM (DUMMY);
         PUT EDIT(DUMMY)(SKIP(1),A(208));
         GO TO RD;
  STOP:  CLOSE FILE (SUB);


  /*  THIS PROCEDURE TAKES A STRING (-BREAKE-) AND BREAKS IT UP INTO
      PIECES OF LENGTH -LNTH-.  THE PIECES ARE STORED IN -BROKE- AND
      -NU- INDICATES HOW MANY PIECES IT TOOK.                          */


  BREAK:  PROCEDURE(BREAKE,LNTH,BROKE,NU);
          DECLARE
          BREAKE            CHARACTER (396) VARYING,
          LNTH              FIXED BINARY (31,0),
          TEMP              CHARACTER (396) VARYING,
          BROKE(14)         CHARACTER (WIDTH) VARYING,
          NU                FIXED BINARY (31,0),
          L                 FIXED BINARY (31,0),  .
          SUM1              FIXED BINARY (31,0),
        ;
        NU = 0:
INIT:   SUM1 = 0:
        TEMP = BREAKE:
SRCH:   L = INDEX(TEMP,' '):
        IF L = 0
          THEN DO:
               L = LENGTH(TEMP);
               SUM1 = SUM1 + L;
               IF SUM1 > LNTH
                 THEN DO:
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L-1);
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,SUM1-L+1):
                      RETURN;
                      END:
                 ELSE DO:
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,1,SUM1);
                      RETURN;
                      END;
               END;
        IF L > LNTH
          THEN DO:
               PUT FILE (ERRE) EDIT
                 ('TROUBLE IN BREAK WITH ',VT_NO)
                 (SKIP(2),A(22),A(10));
               GO TO READ;
               END;
SUM:    SUM1 = SUM1 + (L - 1);
        IF SUM1 <=LNTH THEN DO;
                            TEMP = SUBS TR(TEMP,L+1):
                            SUM1 = SUM1 + 1;
                            GO TO SRCH;
                            END;
        NU = NU + 1;
        BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L);
        BREAKE = SUBSTR(BREAKE,SUM1+2-L);
        SUM1 = 0;
        GO TO SUM;
        END BREAK;

        END WORK;

        CLOSE FILE (ERRE), FILE (SORTOUT);
        END SUBIND;
```

AUTHOR QUARTERLY

```
FIRST: PROCEDURE OPTIONS(MAIN);
 /*********************************************************************/
 /*                                                                   */
 /*           Q U A R T E R L Y     A U T H O R     I N D E X         */
 /*                                                                   */
 /*                                                                   */
 /*                                                                   */
 /* PURPOSE: EXTRACT DATA FROM MASTER TAPE TO BE USED FOR QUARTERLY   */
 /*           INDEX, ANNUAL INDEX, AND DATA BASE TO SIMPLIFY FINDING  */
 /*           PUBLICATION INFORMATION REGARDING A GIVEN AUTHOR.       */
 /*           NOTE: AUTHOR'S NAMES ARE CAPITALIZED IN THE OUTPUT DATA */
 /*           VARIABLES REQUIRING DEFINITIONS ARE DEFINED WITHIN THE  */
 /*           PROGRAM.                                                */
 /*                                                                   */
 /*********************************************************************/
DCL BUFFER CHAR(3736) VAR,
     IHESRTA
        ENTRY(CHAR(48),CHAR(34),FIXED BINARY(31,0),FIXED BINARY(15,0)),
        RETCDE FIXED BINARY(15,0),
        SFIELD CHAR(48),
        RFIELD CHAR(34),
      BEGIN_SHORT CHAR(1) INITIAL('<'),
      END_SHORT CHAR(1) INITIAL('>'),
       SHORT_TITLE  CHAR(396) VARYING,
     VT# CHAR(10),
     ED# CHAR(20),
     NAME CHAR(132),
      UPC CHAR(132) INITIAL((132)' '),
     RESUME_PG CHAR(4),
      PER_AUT CHARACTER(132),
      TITLE CHARACTER(396) VARYING,
      CORP_AUT CHARACTER(528) VARYING,
     (I,#_PER_AUT,LEN_P_AUT(2),TITLE_LEN,#_COR_AUT,LEN_C_AUT(5),#SA,
          LSA(2),J,LP1,LC,LG,LPA,LP2,LB,LEN_AVAIL,LEN_J_C,LEN_PUB,
         LEN_PG#,J1,J2) FIXED BINARY(31,0),
      PUB_DATE CHARACTER(100) VARYING,
      PAGE CHARACTER(10) VARYING,
     BUF2 CHAR(560) VAR;
  /*
         CREATE  AUTHOR  FILE
  */
CREATE: BEGIN;
  OPEN FILE(SYSPRINT) LINESIZE(120);
OPEN FILE(PERSON) TITLE('SORTIN') RECORD OUTPUT;
  ENTER: READ FILE(MASTER) INTO(BUFFER);
   ON ENDFILE(MASTER) GOTO FINISH;
         LEN_C_AUT=0 ;
         LEN_P_AUT=0 ;
        LSA=0;
  /*
         FIND BEGINNING OF LENGTHS
  */
           I=INDEX(BUFFER,'||');
         IF I>0 THEN GO TO AUTHOR;
                 ELSE DO;
  /* SIGNAL FOR LENGTHS IS MISSING - TERMINATE */
                        PUT LIST('INDEX IS WRONG',I);
                        GO TO FINISH; END;
 AUTHOR: ;
  /* FIND PER. AUTHORS LENGTH */
      GET STRING(BUFFER) EDIT(#_PER_AUT)(X(I+2),F(4));
         I=I+6;
          IF #_PER_AUT>2 THEN DO;
```

```
                                      PUT EDIT(BUFFER)(SKIP,A);
              PUT LIST('  # PERSONAL AUTHORS WRONG',#_PER_AUT);
         #_PER_AUT=0:   END;
         IF #_PER_AUT=2 THEN GET STRING(BUFFER) EDIT(LEN_P_AUT(1),
             LEN_P_AUT(2))(X(I),2 F(4)); ELSE
         IF #_PER_AUT=1 THEN GET STRING(BUFFER) EDIT(LEN_P_AUT(1))
             (X(I),F(4));
    I= I+#_PER_AUT*4;
 /* FIND TITLE LENGTH AND CORP. AUTHORS */
    GET STRING(BUFFER) EDIT(TITLE_LEN,#_COR_AUT)(X(I),2 F(4));
       I= I+8;
        IF #_COR_AUT>5 THEN DO;
                                      PUT EDIT(BUFFER)(SKIP,A);
              PUT LIST('  # CORP. AUTHOR WRONG',#_COR_AUT);
              #_COR_AUT=0;      END;
     IF #_COR_AUT>0 THEN
        GET STRING(BUFFER) EDIT((LEN_C_AUT(J)  DO J = 1 TO #_COR_AUT))
             (X(I), 5 F(4));
I= I+#_COR_AUT*4;
 /* FIND SPACE TO AVAILABILITY INFO. */
        GET STRING(BUFFER) EDIT(#SA)(X(I),F(4));
       I= I+4;
    IF #SA>3 THEN DO:
                                      PUT EDIT(BUFFER)(SKIP,A);
              PUT LIST('  SPON. AG. WRONG',#SA);
                   #SA=0;   END;
     IF #SA>0 THEN
        GET STRING(BUFFER) EDIT((LSA(J)  DO J= 1 TO #SA))
       (X(I),3 F(4));
I= I+#SA*4;
        GET STRING(BUFFER) EDIT(LP1,LC,LG,LPA,LP2,LB,LEN_AVAIL,LEN_J_C
         ,LEN_PUB,LEN_PG#)(X(I),10 F(4));
 /* FIND DATA FOR PERSONAL AUTHOR RECORD AND PLACE IN FILE */
 /*
        I IS THE LENGTH OF INFORMATION TO BE SKIPPED
 */
       I=LSA(1)+LSA(2)+LP1+LC+LG+LPA+LP2+LB+LEN_AVAIL+LEN_J_C;
    GET STRING(BUFFER) EDIT(VT#,ED#,RESUME_PG,PER_AUT,TITLE,CORP_AUT,
       PUB_DATE)(A(10),X(17),A(20),X(7),A(4),A(SUM(LEN_P_AUT)),
       A(TITLE_LEN),A(SUM(LEN_C_AUT)),X(I),A(LEN_PUB));
    PUB_DATE=SUBSTR(PUB_DATE,6,2);
/* PULL OFF SHORT TITLE */
        J2=0;
 PULL:  J= INDEX(TITLE,BEGIN_SHORT):
        IF J=0 THEN GO TO PUTOUT:
         IF SUBSTR(TITLE,J+1,1)=BEGIN_SHORT THEN J=J+1:
         J1= INDEX(TITLE,END_SHORT);
         IF J1=0 THEN GO TO PUTOUT;
       SUBSTR(TITLE,J1,1)=SUBSTR(TITLE,J1+1,1);
         J1= J1+1;
 IF J2=0 THEN SHORT_TITLE=SUBSTR(TITLE,J+1,J1-J-1): ELSE
         SHORT_TITLE=SHORT_TITLE||' '||SUBSTR(TITLE,J+1,J1-J-1);
         TITLE= SUBSTR(TITLE,J1+1);
         J2= J2+1;
         GO TO PULL;
 PUTOUT: IF J2>0 THEN DO;
               TITLE=SHORT_TITLE;
               TITLE_LEN=LENGTH(SHORT_TITLE); END;
      IF #_PER_AUT¬=0 THEN DO;
 /*
         SET UP RECORD FOR AUTHOR FILE
 */
```

```
          NAME=SUBSTR(PER_AUT,1,LEN_P_AUT(1));
          UNSPEC(NAME)=UNSPEC(NAME)|UNSPEC(UPC);
  /* * *   MAKE UP FIRST PERSONAL AUTHOR RECORD   * * */
       PUT STRING(BUF2) EDIT(NAME,VT#,ED#,RESUME_PG,PUB_DATE,TITLE_LEN,
           TITLE)(A(132),A(10),A(20),A(4),A(2),F(3),A(TITLE_LEN));
        WRITE FILE(PERSON) FROM(BUF2); END;
      IF #_PER_AUT=2 THEN DO;
  /*  PLACE NEXT PERSONAL AUTHOR IN RECORD */
          NAME=SUBSTR(PER_AUT,LEN_P_AUT(1)+1,LEN_P_AUT(2));
          UNSPEC(NAME)=UNSPEC(NAME)|UNSPEC(UPC);
          SUBSTR(BUF2,1,132)=NAME;
        WRITE FILE(PERSON) FROM(BUF2); END;
       IF #_COR_AUT¬=0 THEN DO;
          NAME=SUBSTR(CORP_AUT,1,LEN_C_AUT(1));
          UNSPEC(NAME)=UNSPEC(NAME)|UNSPEC(UPC);
  /* * *   MAKE UP FIRST INSTITUTIONAL AUTHOR RECORD   * * */
        PUT STRING(BUF2) EDIT(NAME,VT#,ED#,RESUME_PG,PUB_DATE,TITLE_LEN,
           TITLE)(A(132),A(10),A(20),A(4),A(2),F(3),A(TITLE_LEN));
        WRITE FILE(PERSON) FROM(BUF2); END;
        LC=1;
        IF #_COR_AUT>1 THEN DO J=2 TO #_COR_AUT;
          LC=LC+LEN_CAUT(J-1);
          NAME=SUBSTR(CORP_AUT,LC,LEN_C_AUT(J));
          UNSPEC(NAME)=UNSPEC(NAME)|UNSPEC(UPC);
  /*  PLACE NEXT INSTITUTIONAL AUTHOR RECORD */
          SUBSTR(BUF2,1,132)=NAME;
          WRITE FILE(PERSON) FROM(BUF2); END;
           GO TO ENTER;
  /*
          CLOSE FILES IN PREPARATION FOR SORT
   */
 FINISH: CLOSE FILE(MASTER), FILE(PERSON);
 END CREATE;
 SFIELD=' SORT FIELDS=(5,132,CH,A,137,10,CH,A),SIZE=E300 ';
 RFIELD=' RECORD TYPE=V,LENGTH=(564,,,171) ';
 CALL IHESRTA(SFIELD,RFIELD,45056,RETCDE);
    /*    */
    /*  END OF CREATION STEP  */
    /*    */
 IF RETCDE=0 THEN BEGIN;
  /*     IF SORT IS NOT SUCCESSFUL SKIP THIS BLOCK, OTHERWISE PRINT INDE
  */
 DCL (LNA,LN_COL,LEN_HEADING,K,PRINT_LINE,QUARTER,PG_L_OR_R,NEXT_INITIAL
      ,VT_ED_SKIP
     ,MARGIN,#_COLS,COL_COUNT,INDEX_PAGE,II,LEN_PAGE,INITIAL_LINE)
          FIXED BINARY(31,0),
     TOP CHAR(4),
     HEADING CHAR(20) VAR,
     COL_SPACE CHAR(20) VAR,
     DATE CHAR(2),
     PREVIOUS CHAR(132) INITIAL((132)' ');
  /*
     OBTAIN PARAMETERS FOR THIS PRINT
           #_COLS = NUMBER OF PRINT COLUMNS
           MARGIN = TOTAL PRINT SPACE IN COLUMN FOR AUTHORS NAME
           LEN_PAGE = NUMBER OF PRINT LINES
           HEADING = WHICH QUARTERLY INDEX AND YEAR FOR EXAMPLE - AIM/FAL
           TOP = INDICATOR FOR POSITIONING OF HEADING AND INDEX PAGE NUMB
                 IF TOP='LEFT'THEN INDEX PAGE NUMBER APPEARS ON THE LEFT
                 AND HEADING APPEARS ON RIGHT
                 IF TOP='RITERREVERSE WILL OCCUR
           COL_SPACE = CHARACTER STRING OF BLANKS TO BE PLACED BETWEEN CO
```

```
         VT_ED_SKIP = NUMBER OF COLUMNS TO BE SKIPPED BETWEEN VT# AND E
              IN VT,ED,RESUME PG. LINE.
         INDEX_PAGE = INITIAL LINE PRINTED IN THIS INDEX
         INITIAL_LINE = LINE FOR 'PAGE' LINE ON TITLE PAGE
         NEXT_INITIAL = LINE FOR 'PAGE' LINE ON PAGES AFTER TITLE PAGE
 */
 /* INITIALIZE PARAMETERS */
GET FILE(SYSIN) DATA(#_COLS,MARGIN,LEN_PAGE,HEADING,TOP,COL_SPACE,
    VT_ED_SKIP,INDEX_PAGE,INITIAL_LINE,NEXT_INITIAL) COPY:
I=LENGTH(COL_SPACE):
LEN_HEADING=#_COLS*(MARGIN+I+3)+8-I;
PRINT_LINE=LEN_HEADING+1;
BEGIN;
DCL (AUTHOR_ARRAY(6),CHAR_COL(10),SUBNAME,SKIP_LINE,ONE_LINE)
        CHAR(MARGIN+3),
    NEW_BLOCK FIXED BINARY(31,0) INITIAL(0),
   PAGE_TOTAL FIXED BINARY(31,0) INITIAL(0);
DCL PRINTER_PAGE(LEN_PAGE) CHAR(LEN_HEADING) VAR;
    DCL AUTHOROUT CHAR(140);
PAGE_TOTAL=0:
SKIP_LINE=' ';
LEN_HEADING=LENGTH(HEADING);
PRINTER_PAGE='          ';
COL_COUNT=1;
QUARTER=PRINT_LINE;
J1=QUARTER-LEN_HEADING;
J2=QUARTER-3;
IF TOP='LEFT' THEN DO;
                    QUARTER=J1;
                    PG_L_OR_R=9;
                    TOP='RITE';
                    END:
               ELSE DO:
                    QUARTER=9;
                    PG_L_OR_R=J2;
                    TOP='LEFT';
                    END;
OPEN FILE(PERSON) RECORD INPUT TITLE('SORTOUT');
OPEN FILE(OUT) PRINT PAGESIZE(LEN_PAGE);
PUT FILE(OUT) PAGE EDIT(INDEX_PAGE,HEADING)(SKIP,CCLUMN(PG_L_OR_R),F(3)
      ,SKIP(0),COLUMN(QUARTER),A(LEN_HEADING)):
PUT FILE(OUT) LINE(INITIAL_LINE) EDIT(('PAGE' DO II=1 TO #_COLS))
     (X(MARGIN+7),A(4),2 (X(MARGIN+I-1),A(4)));
ON ENDPAGE(OUT) BEGIN:
       DO I=1 TO LEN_PAGE-INITIAL_LINE:
              PUT FILE(OUT) EDIT(PRINTER_PAGE(I))(SKIP,A);
        END:
               INDEX_PAGE=INDEX_PAGE+1;
        PRINTER_PAGE='          ';
        PAGE_TOTAL=0:
        IF TOP='LEFT' THEN DO;
                             TOP='RITE';
                             PG_L_OR_R=9;
                             QUARTER=J1;
                          END;
                      ELSE DO:
                             PG_L_OR_R=J2;
                             QUARTER=9;
                             TOP='LEFT';
                           END;
        IF LEN_PAGE<=66 THEN PUT FILE(OUT) PAGE;
                      PUT FILE(OUT) PAGE LINE(2);
                      PUT FILE(OUT) EDIT(INDEX_PAGE,HEADING,
                        'PERSONAL AND INSTITUTIONAL AUTHOR INDEX',
                        ('PAGE' DO II=1 TO #_COLS))(COLUMN(PG_L_OR_R),
                       F(3),SKIP(0),COLUMN(QUARTER),A(LEN_HEADING),
                         SKIP(2),
                         X(PRINT_LINE/2-20),A(39),SKIP(3),X(MARGIN+7),
                         A(4),2 (X(MARGIN+LENGTH(COL_SPACE)-1),A(4)));
                       COL_COUNT=1;
                        INITIAL_LINE=NEXT_INITIAL;
                            END;
ON CONVERSION BEGIN;
PUT FILE(SYSPRINT) LIST('** CONVERSION ERROR ** REC= ',BUF2,' **');
GO TO IN;
END;
              ON ENDFILE(PERSON) BEGIN;
                       SIGNAL ENDPAGE(OUT);
                       GO TO FINI;
                                  END;
IN: READ FILE(PERSON)INTO(BUF2);
LNA=0;
   GET STRING(BUF2) EDIT(NAME,VT#,ED#,RESUME_PG,DATE,TITLE_LEN,TITLE)
        (A(132),A(10),A(20),A(4),A(2),F(3),A(TITLE_LEN));
    /* SHORTEN AND STORE AUTHOR INFORMATION FOR ANNUAL */
      AUTHOROUT=NAME||DATE||SUBSTR(VT#,4,3)||SUBSTR(VT#,8,3);
  PUT FILE(ANPRT) EDIT(AUTHOROUT)(SKIP,X(2),A);
        WRITE FILE(ANNUAL) FROM(AUTHOROUT);
    IF NAME=PREVIOUS THEN GO TO VT_ED;
    PREVIOUS=NAME;
LNA=1;
    AUTHOR_ARRAY(LNA)=SKIP_LINE;
 /*
        BREAK NAME UP FOR PRINT
 */
   DO J=MARGIN, 2*MARGIN-I,J+MARGIN-I,J+MARGIN-I,J+MARGIN-I;
    LNA=LNA+1;
     SUBNAME=SUBSTR(NAME,J-MARGIN+1,MARGIN);  I=0;
              IF SUBSTR(SUBNAME,MARGIN,1)¬=' ' THEN DO;
              DO I=1 TO MARGIN-1;
                IF SUBSTR(SUBNAME,MARGIN-I,1)=' ' THEN GO TO BREAK;
                  END;
        PUT LIST(' AUTHOR NAME IS WRONG **',NAME,' **  VT# = ',VT#);
                                          END;
BREAK: SUBNAME=SUBSTR(SUBNAME,1,MARGIN-I);
       K=INDEX(SUBNAME,'  ');
       IF K=1 THEN GO TO VT_ED;
      IF K=0 THEN K=MARGIN-I;
    PUT STRING(ONE_LINE) EDIT(SUBNAME)(A(K));
    AUTHOR_ARRAY(LNA)=ONE_LINE;
   END;
VT_ED: LN_COL=0;
 IF LNA>0 THEN LNA=LNA-1;
 /*
        SET-UP STRING FOR - VT  ED  RESUME PG.  LINE
 */
      PUT STRING(ONE_LINE) EDIT(VT#,ED#,RESUME_PG)(A(10),X(VT_ED_SKIP)
                    ,A(20),X(MARGIN-31-VT_ED_SKIP),A(4));
    LN_COL=LN_COL+1;
     CHAR_COL(LN_COL)=ONE_LINE;
     IF TITLE_LEN<=MARGIN THEN DO;
    PUT STRING(ONE_LINE) EDIT(TITLE)(X(2),A);
    LN_COL=LN_COL+1;
    CHAR_COL(LN_COL)=ONE_LINE;
    GO TO IN1;
                          END;
   K=MARGIN-2;
 /*
       BREAK UP TITLE ALLOW MAXIMUM OF 14 LINES
 */
   DO J=K, 2*K-I,J+K-I,J+K-I,J+K-I,J+K-I,J+K-I,J+K-I,J+K-I,J+K-I,J+K-I
       ,J+K-I,J+K-I,J+K-I;
        IF TITLE_LEN<=J THEN DO;
  PUT STRING(ONE_LINE) EDIT(SUBSTR(TITLE,J-K+1,TITLE_LEN-J+K))(X(2),A);
   LN_COL=LN_COL+1;
   CHAR_COL(LN_COL)=ONE_LINE;
    GO TO IN1;
                             END;
       IF TITLE_LEN<J-K+1 THEN GO TO IN1;
      LAB1: DO I=0 TO K-1;
       IF SUBSTR(TITLE,J-I,1)=' '|SUBSTR(TITLE,J-I,1)='-' THEN DO;
  PUT STRING(ONE_LINE) EDIT(SUBSTR(TITLE,J-K+1,K-I))(X(2),A);
   LN_COL=LN_COL+1;
   CHAR_COL(LN_COL)=ONE_LINE;
         GO TO NEXTJ; END LAB1;
       PUT LIST('TITLE ERROR**',TITLE, '**  VT# = ',VT#);
NEXTJ: END;
IN1: NEW_BLOCK=PAGE_TOTAL+LN_COL+LNA;
 /*
        FILL IN PRINTER PAGE
 */
      IF NEW_BLOCK>LEN_PAGE-INITIAL_LINE THEN DO;
             IF COL_COUNT=1 THEN DO I=PAGE_TOTAL + 1 TO LEN_PAGE;
                 PRINTER_PAGE(I)=PRINTER_PAGE(I)||SKIP_LINE;
                                END;
                           ELSE DO I=PAGE_TOTAL + 1 TO LEN_PAGE;
               PRINTER_PAGE(I)=PRINTER_PAGE(I)||COL_SPACE||SKIP_LINE;
                                END;
                                  COL_COUNT=COL_COUNT+1;
                                  IF COL_COUNT>#_COLS THEN SIGNAL
                                            ENDPAGE(OUT);
                                  IF LNA=0 THEN DO;
                                               PAGE_TOTAL=1;
                                 IF COL_COUNT=1 THEN
                                     PRINTER_PAGE(1)=PRINTER_PAGE(1)||
                                            SKIP_LINE;
                                          ELSE
                                     PRINTER_PAGE(1)=PRINTER_PAGE(1)
                                        ||COL_SPACE||SKIP_LINE;
                                              END;
                                           ELSE PAGE_TOTAL=0;
                                END;
                       II=PAGE_TOTAL;
                       IF LNA>0 THEN DO I=1 TO LNA;
                            II=PAGE_TOTAL+I;
             IF COL_COUNT=1 THEN PRINTER_PAGE(II)=PRINTER_PAGE(II)
                         ||AUTHOR_ARRAY(I); ELSE
                      PRINTER_PAGE(II)=PRINTER_PAGE(II)||COL_SPACE||
                               AUTHOR_ARRAY(I);
                                     END;
                       DO I=1 TO LN_COL;
                          II=II+1;
             IF COL_COUNT=1 THEN PRINTER_PAGE(II)=PRINTER_PAGE(II)
                         ||CHAR_COL(I); ELSE
                      PRINTER_PAGE(II)=PRINTER_PAGE(II)||COL_SPACE||
                               CHAR_COL(I);
```

```
                                   END;                                   00003740
                                   PAGE_TOTAL=II;                         00003750
   GO TO IN;                                                              00003760
   FINI: CLOSE FILE(PERSON), FILE(OUT);                                   00003770
   PUT FILE(SYSPRINT) DATA(INDEX_PAGE);                                   00003780
     /*  */                                                               00003790
    /*   END OF LIST ROUTINE  */                                          00003800
     /*  */                                                               00003810
   END;                                                                   00003820
        END FIRST;                                                        00003830
```

SUPPORTING SERVICES QUARTERLY

```
/**********************************************************************/00000010
/*                                                                    */00000020
/*  Q U A R T E R L Y   S U P P O R T I N G   S E R V I C E S        */00000030
/*          I N D E X    P R O G R A M                                */00000040
/*                                                                    */00000050
/*                                                                    */00000060
/*                                                                    */00000070
/*  PURPOSE:  THE PURPOSE OF THIS PROGRAM IS TO PRODUCE THE           */00000070
/*            QUARTERLY SUPPORTING SERVICES  INDEX  FOR BOTH AIM OR   */00000080
/*            ARM.  THE MAIN INPUT IS THE MASTER TAPE DATA SET FOR    */00000090
/*            THE QUARTER.  A DISK SUBSET FILE IS CREATED FROM        */00000100
/*            MASTER AND THEN THE FILE IS SORTED TO ANOTHER DISK      */00000110
/*            FILE USING THE 360 SORT/MERGE.  THE INDEX PRINTOUT      */00000120
/*            COMES FROM PRINTING THE SORTED FILE IN A FORMAT         */00000130
/*            SPECIFIED BY THE VOCATIONAL CENTER ON AN OUTPUT FILE,   */00000140
/*            WHICH IS LATER LISTED ON THE 360/50 FOR UPPER AND       */00000150
/*            LOWER CASE PRINTING.  THE SORTED FILE IS SHORTENED      */00000160
/*            AND SAVED ON TAPE FOR THE ANNAUL INDEX.                 */00000170
/*                                                                    */00000180
/*  DEFINITION OF VARIABLES:                                          */00000190
/*      1. AFIELD -- CONVERTED 3-DIGIT SUB-CODE.                      */00000200
/*      2. BT -- DESIGNATES BEGIN SHORT TITLE SEGMENT CHARACTER (<).  */00000210
/*      3. COL -- NUMBER OF COLUMNS ON AN INDEX PRINTOUT PAGE.        */00000220
/*      4. DUMMY -- STRING USED TO READ AND WRITE FILES.              */00000230
/*      5. ED-NO -- ACCESSION NUMBER.                                 */00000240
/*      6. ET -- DESIGNATES END SHORT TITLE SEGMENT CHARACTER (>).    */00000250
/*      7. FIELD -- ORIGINAL SUB-CODE FROM MASTER IN 2-DIGIT FORM.    */00000260
/*      8. HEAD -- STRING CONTAINING AIM OR ARM, SEASON, AND YEAR.    */00000270
/*                 IT APPEARS ON THE LINE WITH THE INDEX PAGE NUMBER, */00000280
/*                 BUT AT OPPOSITE SIDES OF THE PAGE.                 */00000290
/*      9. IHESRTA -- CALL TO SORT/MERGE.                             */00000300
/*     10. LINEH1 -- TITLE PAGE LINE NUMBER ON WHICH PAGE NUMBER AND  */00000310
/*                   -HEAD- APPEAR.                                   */00000320
/*     11. LINEH2 -- SAME AS LINEH1 BUT FOR SUCCEEDING PAGES.         */00000330
/*     12. LINEP1 -- TITLE PAGE LINE NUMBER ON WHICH THE WORD 'PAGE'  */00000340
/*                   APPEARS ABOVE EACH COLUMN.                       */00000350
/*     13. LINEP2 -- SAME AS LINEP1 BUT FOR SUCCEEDING PAGES.         */00000360
/*     14. LINET -- LINE NUMBER ON ALL PAGES BUT TITLE PAGE ON WHICH  */00000370
/*                  THE TITLE OF THE INDEX APPEARS.  THERE IS NO TITLE*/00000380
/*                  ON THE FIRST PAGE.                                */00000390
/*     15. LENTH -- DATA LENGTHS READ FROM A MASTER RECORD AND USED   */00000400
/*                  IN THE POSITIONING AND RETRIEVING OF NEEDED DATA. */00000410
/*     16. LH -- LENGTH OF VARIABLE -HEAD-.                           */00000420
/*     17. LTH -- LENGTH OF VARIABLE -TITL-.                          */00000430
/*     18. L_TITLE -- LENGTH OF VARIABLE -TITLE-.                     */00000440
/*     19. MARG -- LEFT-HAND MARGIN OF THE INDEX PAGE.                */00000450
/*     20. MASTER -- NAME OF MASTER TAPE.                             */00000460
/*     21. MG -- ARRAY OF COLUMN MARGINS COUNTED FROM COLUMN 1.       */00000470
/*     22. NUMB -- DATA LENGTH READ FROM MASTER AND USED IN THE       */00000480
/*                 POSITIONING OF AND RETRIEVING OF OTHER DATA FROM   */00000490
/*                 MASTER.                                            */00000500
/*     23. ONE -- SUB-CODE OF '1' REPRESENTING 'TECHNICAL EDUCATION'. */00000510
/*     24. PAGE -- PAGE NUMBER OF INDEX PRINTOUT.                     */00000520
/*     25. PAGE_NO -- RESUME PAGE FROM MASTER RECORD.                 */00000530
/*     26. PUB_DATE -- PUBLICATION DATE FROM MASTER RECORD.           */00000540
/*     27. QUE -- INDICATES WHETHER THE PAGE NUMBER ON THE TITLE PAGE */00000550
/*                IS IN THE LEFT ('L') OR RIGHT-HAND ('R') MARGIN.    */00000560
/*                IT REVERSES ITSELF FOR SUCCEEDING PAGES AND -HEAD-  */00000570
/*                IS ALWAYS ON THE OPPOSITE SIDE.                     */00000580
/*     28. RETCDE -- CODE RETURNED BY SORT; '0' IS GOOD AND '16' IS   */00000590
/*                   BAD.                                             */00000600
/*     29. RFIELD -- DEFINES THE INPUT RECORDS FOR SORT.              */00000610
/*     30. SFIELD -- DEFINES WHAT PART(S) OF THE RECORDS WE ARE       */00000620
```

```
                SORTING UNDER.                                              */00000630
  /*   31. SZ   -- RIGHT-HAND MARGIN FOR INDEX PRINT-OUT PAGE.              */00000640
  /*   32. SORTIN -- NAME OF UNSORTED DISK SUBSET FILE.                     */00000650
  /*   33. SORTOUT -- NAME OF SORTED DISK SUBSET FILE.                      */00000660
  /*   34. SPACE -- NUMBER OF SPACES LEFT BETWEEN VT AND ED NUMBER          */00000670
  /*                WHENEVER THEY ARE PRINTED IN THE INDEX.                 */00000680
  /*                                                                        */00000690
  /*   35. SUM1 -- NUMBER OF POSITIONS ON A MASTER RECORD BETWEEN           */00000700
  /*               RESUME PAGE AND TITLE.                                   */00000710
  /*   36. SUM2 -- NUMBER OF POSITIONS ON A MASTER RECORD BETWEEN           */00000720
  /*               TITLE AND PUBLICATION DATE.                              */00000730
  /*   37. SIZE -- MAXIMUM LENGTH OF INDEX PRINT PAGE.                      */00000740
  /*   38. TITL -- TITLE OF THE INDEX.                                      */00000750
  /*   39. TITLE -- TITLE OF A DOCUMENT FROM A MASTER RECORD.               */00000760
  /*   40. TLE -- ARRAY OF SHORT TITLE SEGMENTS FROM THE TITLE.             */00000770
  /*   41. VT_NO -- DOCUMENT NUMBER FROM MASTER.                            */00000780
  /*   42. WIDTH -- COLUMN WIDTH IN INDEX PRINTOUT.                         */00000790
  /*   43. BLK -- NUMBER OF PIECES OF LENGTH -WIDTH- INTO WHICH A           */00000800
  /*              SUB-CODE NAME IS BROKEN.                                  */00000810
  /*   44. BLKT -- NUMBER OF PIECES OF LENGTH -WIDTH-2- INTO WHICH A        */00000820
  /*               TITLE IS BROKEN.                                         */00000830
  /*   45. CODE_NO -- CURRENT SUB-CODE BEING PRINTED IN THE INDEX.          */00000840
  /*   46. COL# -- COLUMN SETUP FOR A PAGE IN THE INDEX PRINTOUT.           */00000850
  /*               THE FIRST SUBSCRIPT IS THE COLUMN NUMBER ON THE          */00000860
  /*               PAGE, AND THE SECOND SUBSCRIPT IS THE LINE NUMBER        */00000870
  /*               IN THE COLUMN.                                           */00000880
  /*   47. FFIELD -- CONVERTED SUB-CODE ON SUBSET FILE.                     */00000890
  /*   48. FLD -- ARRAY OF SUB-CODE NAMES.                                  */00000900
  /*   49. IMAX -- NUMBER OF COLUMNS PLUS ONE.                              */00000910
  /*   50. IMP -- INDICATES START OF NEW SUB-CODE IN INDEX PRINTOUT.        */00000920
  /*   51. IND -- COLUMN NUMBER BEING FILLED IN AN INDEX PAGE.              */00000930
  /*   52. KNT -- NUMBER OF LINES IN A BLOCK OF DATA.  IT INCLUDES          */00000940
  /*              ONE LINE FOR VT NO., ED NO., AND RESUME PAGE: PLUS        */00000950
  /*              LINES FOR THE TITLE GIVEN BY -BLKT-: PLUS, IF             */00000960
  /*              IMP = 1, LINES FOR THE NEW SUB-CODE NAME GIVEN            */00000970
  /*              BY BLK + 1 (1 IS FOR THE BLANK LINE BETWEEN               */00000980
  /*              DIFFERENT SUB-CODE SECTIONS).                             */00000990
  /*   53. LN -- LINE NUMBER ON AN INDEX PAGE BEING FILLED.                 */00001000
  /*             (1 < LN < SIZE) THIS IS A CHECK TO MAKE SURE WE DO         */00001010
  /*             NOT EXCEED THE PAGE LENGTH.                                */00001020
  /*   54. NN -- LINE NUMBER IN A COLUMN CURRENTLY BEING FILLED.            */00001030
  /*             THIS IS A SUBSCRIPT FOR -COL#-.                            */00001040
  /*   55. OUTS -- OUTPUT FILE FOR INDEX PRINTOUT.                          */00001050
  /*   56. PC -- PIECES OF BROKEN UP SUB-CODE NAME.                         */00001060
  /*   57. PUB -- YEAR PART OF PUBLICATION DATE.                            */00001070
  /*   58. SKPH -- COLUMN WHERE -HEAD- IS BEGUN   TO BE PRINTED.            */00001080
  /*   59. SKPP -- COLUMN IN WHICH THE PAGE NUMBER PRINTING IS STARTED*/00001090
  /*   60. SUB -- NAME OF TAPE WHERE SHORTENED RECORDS ARE SAVED.           */00001100
  /*   61. TLP -- PIECES OF BROKEN UP TITLE.                                */00001110
  /*                                                                        */00001120
  /* INPUT TO THE PROGRAM:                                                  */00001130
  /*   MASTER TAPE DATA SET                                                 */00001140
  /*     DEFINED BY DD CARD IN THE JCL.                                     */00001150
  /*   -SYSIN-, INPUT ON CARDS                                              */00001160
  /*     THIS INPUT IS ONE CONTINUOUS STREAM ON CARDS.  THE VALUES          */00001170
  /*     OF THE VARIABLES LISTED BELOW ARE THE DATA ON THE CARDS.           */00001180
  /*     THE ORDER OF THESE VARIABLES IS THEIR ORDER ON THE CARDS           */00001190
  /*     AS WELL, AND THE FORMAT LISTED BELOW FOLLOWS WHAT IS ON THE        */00001200
  /*     CARDS TOO.  THESE VARIABLES ARE DEFINED IN THE 'DEFINITION'        */00001210
  /*     SECTION ABOVE, AND THEIR VALUES CAN BE RECEIVED FROM THE           */00001220
  /*     VOCATIONAL CENTER OR THE TYPED WRITE-UP.                           */00001230
  /*          -PAGE-       F(4,0)                                           */00001240
  /*          -SIZE-       F(4,0)
```

```
/*          -WIDTH-     F(4,0)                                        */
/*          -MARG-      F(4,0)                                        */
/*          -SZ-        F(4,0)                                        */
/*          -LINEH1-    F(4,0)                                        */
/*          -LINEP1-    F(4,0)                                        */
/*          -LINEH2-    F(4,0)                                        */
/*          -LINET-     F(4,0)                                        */
/*          -LINEP2-    F(4,0)                                        */
/*          -SPACE-     F(4,0)                                        */
/*          -COL-       F(4,0)                                        */
/*          -MG-        (COL) * F(4,0)                                */
/*          -LTH-       F(4,0)                                        */
/*          -TITL-      A(LTH)                                        */
/*          -LH-        F(4,0)                                        */
/*          -HEAD-      A(LH)                                         */
/*          -QUE-       A(1)                                          */
/*                                                                    */
/*  SPECIAL NOTE:                                                     */
/*    THE OUTPUT FILE SHOULD BE THE PRINTER UNTIL A GOOD INDEX        */
/*    PRINTOUT IS OBTAINED AND THE VOCATIONAL CENTER HAS APPROVED     */
/*    IT.  THEN THE OUTPUT FILE CAN BE CHANGED TO A TAPE AND ONCE     */
/*    THE TAPE IS LISTED ON THE 360/50 WITH THE TN CHAIN, WE HAVE     */
/*    THE UPPER AND LOWER CASE LETTERS IN THE PRINTOUT.               */
/*                                                                    */
/**********************************************************************/


VOCIND: PROCEDURE OPTIONS (MAIN);
        DECLARE
           LINEH1            FIXED BINARY (31,0),
           L
               FIXED BINARY (31,0),
           AFIELD
               CHARACTER (3),
           BT                CHARACTER (1) INITIAL ('<'),
           COL
               FIXED BINARY (31,0),
           DUMMY             CHARACTER (3736) VARYING,
           ED_NO             CHARACTER (20),
           ET                CHARACTER (1) INITIAL ('>'),
           FIELD
               CHARACTER (2),
           HEAD              CHARACTER (13) VARYING,
           I
               FIXED BINARY (31,0),
           IHESRTA           ENTRY (CHARACTER (45),CHARACTER (33),
                             FIXED BINARY (31,0),FIXED BINARY (15,0)),
           J
               FIXED BINARY (31,0),
           LINEH2            FIXED BINARY (31,0),
           LINEP1            FIXED BINARY (31,0),
           LINEP2            FIXED BINARY (31,0),
           LINET             FIXED BINARY (31,0),
           LENTH(10)         FIXED BINARY (31,0),
           LH                FIXED BINARY (31,0),
           LTH
               FIXED BINARY (31,0),
           L_TITLE           FIXED BINARY (31,0),
           M
               FIXED BINARY (31,0),
           MARG              FIXED BINARY (31,0),
           MASTER
```

```
       FILE SEQUENTIAL UNBUFFERED INPUT RECORD,
     MG(3)          FIXED BINARY (31,0),
     N
       FIXED BINARY (31,0),
     NUMB           FIXED BINARY (31,0),
     ONE
       CHARACTER (1),
     PAGE
       FIXED BINARY (31,0),
     PAGE_NO        CHARACTER (4),
     PUB_DATE
       CHARACTER (7),
     QUE            CHARACTER (1),
     RETCDE
       FIXED BINARY (15,0),
     RFIELD         CHARACTER (33),
     SFIELD         CHARACTER (45),
     SIZE
       FIXED BINARY (31,0),
     SORTIN
       FILE SEQUENTIAL RECORD,
     SORTOUT
       FILE SEQUENTIAL RECORD,
     SPACE          FIXED BINARY (31,0),
     SUM1
       FIXED BINARY (31,0),
     SUM2
       FIXED BINARY (31,0),
     SZ             FIXED BINARY (31,0),
     TITL
       CHARACTER (50) VARYING,
     TITLE          CHARACTER (396) VARYING,
     TLE(3)
       CHARACTER (396) VARYING,
     VT_NO
       CHARACTER (10) VARYING,
     WIDTH
       FIXED BINARY (31,0),
    ;

    OPEN FILE (MASTER),
       FILE (ERRE) OUTPUT PRINT STREAM LINESIZE (125),
       FILE (SORTIN) OUTPUT;
    ON ENDFILE (MASTER) GO TO SORT;


 /* READ A RECORD FROM THE MASTER TAPE DATA SET.  FIND THE DIVIDER IN
    THE RECORD.                                                        */


READ_MASTER:
    READ FILE (MASTER) INTO (DUMMY);
    N = INDEX(DUMMY,'||');
    IF N = 0
     THEN DO;
         VT_NO = SUBSTR(DUMMY,1,10);
         PUT FILE (ERRE) EDIT
           ('MISSING DIVIDER IN DOCUMENT ',VT_NO)
           (SKIP(2),A(28),A(10));
         GO TO READ_MASTER;
         END;

    /*  FIND THE NUMBER OF POSITIONS ON THE TAPE BETWEEN THE RESUME PAGE
        AND THE TITLE.  STORE THIS VALUE IN -SUM1-.  PICK UP THE TITLE
        LENGTH FROM MASTER.                                                */


           SUM1 = 0;
           GET STRING (DUMMY) EDIT (NUMB,(LENTH(I) DO I = 1 TO NUMB),
             L_TITLE)  (X(N+1),F(4),(NUMB+1) F(4));
           N = N + 9 + (NUMB * 4);
           IF NUMB¬= 0 THEN DO;
                            LOOP1: DO I = 1 TO NUMB;
                                   SUM1 = SUM1 + LENTH(I);
                                   END LOOP1;
                            END;


    /*  FIND THE NUMBER OF POSITIONS ON THE TAPE BETWEEN THE TITLE AND
        THE PUBLICATION DATE.  STORE THIS VALUE IN -SUM2-.                 */


           SUM2 = 0;
   LOOP2:  DO J = 1 TO 2;
           GET STRING (DUMMY) EDIT (NUMB,(LENTH(I) DO I=1 TO NUMB))
             (X(N),F(4),(NUMB) F(4));
           N = N + 4 + (4 * NUMB);
           IF NUMB ¬= 0 THEN DO;
                             INNER: DO L = 1 TO NUMB;
                                    SUM2 = SUM2 + LENTH(L);
                                    END INNER;
                             END;
           END LOOP2;
   LOOP3:  DO I = 1 TO 8;
           GET STRING (DUMMY) EDIT (NUMB)
             (X(N),F(4));
           SUM2 = SUM2 + NUMB;
           N = N + 4;
           END LOOP3;


    /*  NOW WITH THE ABOVE INFORMATION, RETRIEVE THE NECESSARY DATA FROM
        MASTER.                                                            */


           GET STRING (DUMMY) EDIT (VT_NO,FIELD,ONE,ED_NO,PAGE_NO,TITLE,
                                    PUB_DATE)
             (A(10),X(5),A(2),A(1),X(9),A(20),X(7),A(4),X(SUM1),A(L_TITLE)
             ,X(SUM2),A(7));
           PUT SKIP (4);
           PUT EDIT (VT_NO,FIELD,ONE,ED_NO,PAGE_NO,TITLE,PUB_DATE) (A);


    /*  FORM THE SHORT TITLE FROM THE TAPE TITLE BY CONCATENATING SHORT
        TITLE SEGMENTS TOGETHER.  A SHORT TITLE SEGMENT STARTS WITH '<'
        AND ENDS WITH '>'.  THE SHORT TITLE IS USED IN THE INDEX.          */


           I = 0;
   RETRN:  N = INDEX(TITLE,BT);
           M = INDEX(TITLE,ET);
           IF (N=0) & (M=0) THEN GO TO LOOP4;
           I = I + 1;
        TLE(I) = SUBSTR(TITLE,N+1,M-N-1);
        IF SUBSTR(TITLF,M+1,1) ¬= ' ' THEN TLE(I) = TLE(I) ||
                                           SUBSTR(TITLE,M+1,1);
        TITLE = SUBSTR(TITLE,M+2);
        GO TO RETRN;
LOOP4:  DO J = 1 TO I;
        IF J = 1 THEN TITLE = TLE(1);
                ELSE TITLE = TITLE || ' ' || TLE(J);
        END LOOP4;
        PUT DATA (TITLE);


 /*  CONVERT THE TWO-DIGIT SUB-CODE TO THREE DIGITS BY ADDING A ZERO
     TO THE RIGHT SIDE.  (EX: 45 BECOMES 450).  THEN FORM A RECORD ON
     THE DISK SUBSET FILE FOR THIS SUB-CODE.                         */


        AFIELD = FIELD || '0';
        PUT STRING (DUMMY) EDIT (AFIELD,VT_NO,ED_NO,PAGE_NO,PUB_DATE,
                                TITLE)
          (A(3),A(10),A(20),A(4),A(7),A);
        WRITE FILE (SORTIN) FROM (DUMMY);
        PUT DATA (AFIELD);


 /*  IF THERE IS A '1' FOR 'TECHNICAL EDUCATION' THEN CONVERT THIS
     SUB-CODE TO '495' AND MAKE A RECORD ON DISK FOR THE SUB-CODE.   */


        IF ONE = '1'
          THEN DO;
               AFIELD = '495';
               PUT STRING (DUMMY) EDIT (AFIELD,VT_NO,ED_NO,PAGE_NO,
                 PUB_DATE,TITLE)
                 (A(3),A(10),A(20),A(4),A(7),A);
               WRITE FILE (SORTIN) FROM (DUMMY);
               PUT DATA (AFIELD);
               END;
        GO TO READ_MASTER;


 /*  SORT THE DISK SUBSET FILE AND PRINT THE RETURN CODE, AN INDICATION
     OF A SUCCESSFUL SORT.                                           */


SORT:   CLOSE FILE (SORTIN),
              FILE (MASTER);
        SFIELD = ' SORT FIELDS=(5,3,FI,A,11,7,CH,A),SIZE=E9999 ';
        RFIELD = ' RECORD TYPE=V,LENGTH=(444,,,50) ';
        CALL IHESRTA(SFIELD,RFIELD,45056,RETCDE);
        PUT FILE (SYSPRINT) EDIT ('RETURN CODE = ',RETCDE)
          (SKIP(3),A(14),F(3));


 /*  READ IN THE PAGE SPECIFICATIONS FOR THE INDEX PRINTOUT.         */


        GET FILE (SYSIN) EDIT (PAGE,SIZE,WIDTH,MARG,SZ,LINEH1,LINEP1,
          LINEH2,LINEFT,LINEP2,SPACE,COL,(MG(N) DO N = 1 TO COL),LTH,
          TITL,LH,HEAD,QUE)
          (13 F(4),(COL) F(4),A(LTH),F(4),A(LH),A(1));

```

```
/*  THIS BLOCK PRINTS THE INDEX ON THE OUTPUT FILE AND SHORTENS AND
    SAVES THE RECORDS ON TAPE.                                       */

WORK:   BEGIN;
        DECLARE
          BLK
              FIXED BINARY (31,0),
          BLKT
              FIXED BINARY (31,0),
          BREAK                ENTRY(,FIXED BINARY (31,0),,),
          CODE_NO
              FIXED DECIMAL (3,0) INITIAL (999),
          COL#(COL,SIZE-LINEP2-1)  CHARACTER (WIDTH+3),
          FFIELD
              FIXED DECIMAL (3,0),
          FLD(0:13)            CHARACTER (50) VARYING INITIAL
                  ('AGRICULTURAL EDUCATION',
                   'BUSINESS AND OFFICE EDUCATION',
                   'DISTRIBUTIVE EDUCATION',
                   'HEALTH OCCUPATIONS EDUCATION',
                   'HOME ECONOMICS EDUCATION',
                   'INDUSTRIAL ARTS EDUCATION',
                   'MANPOWER ECONOMICS',
                   'OCCUPATIONAL PSYCHOLOGY',
                   'OCCUPATIONAL SOCIOLOGY',
                   '',
                   'TRADE AND INDUSTRIAL EDUCATION',
                   '',
                   'GENERAL VOCATIONAL AND TECHNICAL EDUCATION',
                   'TECHNICAL EDUCATION'),
          IMAX
              FIXED BINARY (31,0),
          IMP                  FIXED BINARY (31,0),
          IND
              FIXED BINARY (31,0),
          K
              FIXED BINARY (31,0),
          KK                   FIXED BINARY (31,0),
          KNT                  FIXED BINARY (31,0),
          LL                   FIXED BINARY (31,0),
          LN
              FIXED BINARY (31,0),
          NN                   FIXED BINARY (31,0),
          OUTS
              FILE,
          PC(14)               CHARACTER (WIDTH) VARYING,
          PUB
              CHARACTER (2),
          SKPH                 FIXED BINARY (31,0),
          SKPP                 FIXED BINARY (31,0),
          SUB
              FILE SEQUENTIAL RECORD,
          TLP(14)              CHARACTER (WIDTH) VARYING,
        ;

        ON ENDFILE (SORTOUT) GO TO SHORTEN;
        OPEN FILE (SORTOUT),
            FILE (OUTS) OUTPUT STREAM PRINT PAGESIZE(SIZE+4)
        ;
```

```
 /*  INITIALIZE VARIABLES FOR PRINTING INDEX.                        */


BLKOT:  DO I = 1 TO SIZE-LINEP2-1;
        DO J = 1 TO COL;
        COL#(J,I) = ' ';
        END BLKOT;
        K = PAGE;
        IND =1;
        IMAX = COL + 1;
        LN = LINEP1;
        NN = 0;
        IF QUE = 'L' THEN DO;
                           SKPP = MARG;
                           SKPH = SZ - LH + 1;
                           END;
                      ELSE DO;
                           SKPH = MARG;
                           SKPP = SZ - 3;
                           END;


 /*  PRINT HEADING ON THE TITLE PAGE.                                */


        PUT FILE (OUTS) EDIT (PAGE,HEAD)
          (PAGE,LINE(LINEH1),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
          A(LH));
        PUT FILE (OUTS) SKIP (LINEP1-LINEH1);
        LP8: DO I = 1 TO COL;
             PUT FILE (OUTS) EDIT ('PAGE')
               (SKIP(0),COLUMN(MG(I)+WIDTH-1),A(4));
             END LP8;
        PUT FILE (OUTS) SKIP(1);


 /*  READ A RECORD FROM THE SORTED DISK FILE.  BREAK IT UP INTO ITS
     PIECES OF DATA.  BREAK UP THE TITLE INTO PIECES FITTING IN A
     COLUMN OF OUTPUT.                                               */


READ:   READ FILE (SORTOUT) INTO (DUMMY);
        L = LENGTH(DUMMY);
        GET STRING (DUMMY) EDIT (FFIELD,VT_NO,ED_NO,PAGE_NO,PUB_DATE,
          TITLE)
          (F(3),A(10),A(20),A(4),A(7),A(L-44));
        CALL BREAK (TITLE,WIDTH-2,TLP,BLKT);


 /*  IF A NEW SUB-CODE IS BEING STARTED WITH THIS RECORD, FETCH THE
     SUB-CODE NAME AND BREAK IT UP INTO PIECES FITTING IN A COLUMN
     OF OUTPUT.  ASSIGN '1' TO -IMP- TO INDICATE A NEW SUB-CODE.
     ASSIGN -KNT- THE NUMBER OF LINES OF OUTPUT IN THIS BLOCK OF DATA.
                                                                     */


        IF FFIELD¬=CODE_NO
          THEN DO;
               CODE_NO = FFIELD;
               IF FFIELD = 495 THEN L = 13;
                               ELSE L = (FFIELD - 400)/10;
               IF LENGTH(FLD(L)) > WIDTH
                 THEN DO;
                      TITLE = FLD(L);
                      CALL BREAK (TITLE,WIDTH,PC,BLK);
                      END;
                 ELSE DO;
                      BLK = 1;
                      PC(1) = FLD(L);
                      END;
               KNT = BLK + 2 + BLKT;
               IMP = 1;
               END;


 /*  IF THE SAME SUB-CODE IS CONTINUING, SET -IMP- AND -KNT- TO VALUES
     FOR THIS SITUATION.                                               */


          ELSE DO;
               KNT = BLKT + 1;
               IMP = 0;
               END;


 /*  IF THIS BLOCK OF DATA CANNOT FIT INTO THE COLUMN BEING FILLED
     WITHOUT EXCEEDING THE PAGE LENGTH, START A NEW COLUMN.            */


        IF (LN + KNT) > SIZE THEN GO TO HEADR;


 /*  IF A NEW SUB-CODE IS BEING STARTED, SKIP A LINE IN THE COLUMN AND
     PRINT THE SUB-CODE NAME.                                          */


RETURN: IF IMP = 1 THEN DO;
                        NN = NN + 1;
                        LP6: DO I = 1 TO BLK;
                             COL#(IND,NN) = PC(I);
                             NN = NN + 1;
                             END LP6;
                        END;


 /*  PRINT THE VT#, ED#, AND RESUME PAGE IN THE COLUMN, ALL ON ONE
     LINE.                                                             */


        PUT STRING (COL#(IND,NN)) EDIT (VT_NO,ED_NO,PAGE_NO)
          (A(10),X(SPACE),A(20),X(WIDTH-30-SPACE-1),A(4));
        NN = NN + 1;


 /*  PRINT THE TITLE IN THE COLUMN NEXT.                               */


LP7:    DO I = 1 TO BLKT;
        COL#(IND,NN) = '  ' || TLP(I);
        NN = NN + 1;
        END LP7;
        LN = LN + KNT;
        GO TO READ;
 /*  BRANCH HERE WHEN A COLUMN IS AS FULL AS CAN BE.
     UP THE COLUMN SUBSCRIPT BY ONE.                                      */

HEADR:  IND = IND + 1;

  /*  IF THE LAST COLUMN ON THE PAGE IS FULL, START A NEW PAGE.  FIRST,
      PRINT OUT THE COLUMNS OF THE FINISHED PAGE.  THEN INCREASE THE
      INDEX PAGE AND PRINT OUT A TITLE FOR THE NEW PAGE.                  */


        IF IND = IMAX
          THEN DO;
               IND = 1;
               HD1: DO LL = 1 TO SIZE-LINEP2-1;
                    PUT FILE (OUTS) SKIP(1);
                    LP9: DO KK = 1 TO COL;
                         PUT FILE (OUTS) EDIT (COL#(KK,LL))
                           (SKIP(0),COLUMN(MG(KK)),A(WIDTH+3));
                         COL#(KK,LL) = ' ';
                         END LP9;
                    END HD1;
               PAGE = PAGE + 1;
               IF SKPP =MARG THEN DO;
                                  SKPP = SZ - 3;
                                  SKPH = MARG;
                                  END;
                             ELSE DO;
                                  SKPP = MARG;
                                  SKPH = SZ - LH + 1;
                                  END;
               PUT FILE (OUTS) EDIT (PAGE,HEAD,TITL)
                 (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(
                 SKPH),A(LH),LINE(LINET),X((SZ-LTH-MARG+1)/2+MARG-1),
                 A(LTH));
               PUT FILE (OUTS) SKIP(LINEP2-LINET);
               LP10: DO LL = 1 TO COL;
                      PUT FILE (OUTS) EDIT ('PAGE')
                        (SKIP(0),COLUMN(MG(LL)+WIDTH-1),A(4));
                      END LP10;
               PUT FILE (OUTS) SKIP(1);
               END;

  /*  REINITIALIZE LINE COUNTERS FOR THE NEW COLUMN.                      */


        IF IMP = 1 THEN DO;
                         NN = 0;
                         LN = LINEP2;
                         END;
                    ELSE DO;
                         NN = 1;
                         LN = LINEP2 + 1;
                         END;
        IF K = PAGE THEN LN = LN + LINEP1 - LINEP2;
        GO TO RETURN;
  /*  WHEN ALL RECORDS HAVE BEEN PUT INTO COLUMNS, PRINT THE LAST PAGE
      OF THE INDEX, EVEN THOUGH IT IS NOT COMPLETELY FULL.  WRITE OUT
      THE LAST PAGE NUMBER OF THE INDEX.                                */

 SHORTEN:
           DO I = 1 TO SIZE-LINEP2-1;
           PUT FILE (OUTS) SKIP(1);
           LP11: DO J = 1 TO COL;
                   PUT FILE (OUTS) EDIT (COL#(J,I))
                     (SKIP(0),COLUMN(MG(J)),A(WIDTH+3)):
                   END LP11:
           END SHORTEN;
           PUT FILE (OUTS) PAGE;
           PUT FILE (SYSPRINT) EDIT ('LAST PAGE USED IS ',PAGE)
             (SKIP(3),A(18),F(4));
           CLOSE FILE (OUTS),
                   FILE (SORTOUT);


  /*  SHORTEN THE SORTED RECORDS ON DISK AND PUT THEM ON MAGNETIC TAPE
      FOR THE ANNUAL INDEX.                                             */


           ON ENDFILE (SORTOUT) GO TO STOP;
           OPEN FILE (SORTOUT),
                  FILE (SUB) OUTPUT;
           PUT FILE (SYSPRINT) PAGE;
 RD:       READ FILE (SORTOUT) INTO (DUMMY);
           GET STRING (DUMMY) EDIT (AFIELD,VT_NO,PUB)
             (A(3),A(10),X(29),A(2));
           VT_NO = SUBSTR(VT_NO,4,3) || SUBSTR(VT_NO,8,3):
           PUT STRING (DUMMY) EDIT (AFIELD,PUB,VT_NO,'       ')
             (A(3),A(2),A(6),A(7));
           WRITE FILE (SUB) FROM (DUMMY);
           PUT EDIT (DUMMY) (SKIP(1),A(11));
           GO TO RD;
 STOP:     CLOSE FILE (SUB):


  /*  THIS PROCEDURE BREAKS UP A STRING -BREAKE- INTO PIECES OF LENGTH
      -LNTH- AND STORES THEM IN -BROKE-.  -NU- IS THE NUMBER OF PIECES
      IT TOOK TO DO THE JOB.                                            */


 BREAK:    PROCEDURE(BREAKE,LNTH,BROKE,NU);
           DECLARE
             BREAKE                  CHARACTER (396) VARYING,
             LNTH                    FIXED BINARY (31,0),
             TEMP                    CHARACTER (396) VARYING,
             BROKE(14)               CHARACTER (WIDTH) VARYING,
             NU                      FIXED BINARY (31,0),
             L                       FIXED BINARY (31,0),
             SUM1                    FIXED BINARY (31,0),
           ;
           NU = 0;
 INIT:     SUM1 = 0;
           TEMP = BREAKE;
 SRCH:     L = INDEX(TEMP,' ');
           IF L = 0
             THEN DO;
                   L = LENGTH(TEMP);
                  SUM1 = SUM1 + L;
                  IF SUM1 > LNTH
                    THEN DO;
                          NU = NU + 1;
                          BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L-1);
                          NU = NU + 1;
                          BROKE(NU) = SUBSTR(BREAKE,SUM1-L+1);
                          RETURN;
                          END;
                    ELSE DO;
                          NU = NU + 1;
                          BROKE(NU) = SUBSTR(BREAKE,1,SUM1);
                          RETURN;
                          END;
                  END;
            IF L > LNTH
              THEN DO;
                  PUT FILE (ERRE) EDIT
                    ('TROUBLE IN BREAK WITH ',VT_NO)
                    (SKIP(2),A(22),A(10));
                  GO TO READ;
                  END;
    SUM:    SUM1 = SUM1 + (L - 1);
            IF SUM1 <=LNTH THEN DO;
                                 TEMP = SUBSTR(TEMP,L+1);
                                 SUM1 = SUM1 + 1;
                                 GO TO SRCH;
                                 END;
            NU = NU + 1;
            BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L);
            BREAKE = SUBSTR(BREAKE,SUM1+2-L);
            SUM1 = 0;
            GO TO SUM;
            END BREAK;

            END WORK;

            CLOSE FILE (ERRE), FILE (SORTOUT);
            END VOCIND;
```

DOCUMENT NUMBER QUARTERLY

```
 DOC_INDEX:   PROCEDURE  OPTIONS (MAIN);
 /*
/*  O V E R A L L   P R O G R A M   D E S C R I P T I O N             */
/*                                                                    */
/*                                                                    */
/*    EXTRACT FROM THE MASTER FILE ALL DATA NECESSARY FOR THE         */
/*      DOCUMENT NUMBER INDEX                                         */
/*      TO CREATE THE TITLE TO BE PRINTED                             */
/*    WRITE EACH RECORD ONTO A SORTIN FILE                            */
/*    WHEN THE ENTIRE MASTER FILE HAS BEEN PROCESSED SORT THE SORTIN  */
/*      FILE BY VT NO                                                 */
/*    READ IN THE 2 PRINT CONTROL CARDS WHICH CONTAIN THE DESIRED     */
/*      PRINT SPECIFICATIONS FOR THE DOCUMENT NUMBER INDEX            */
/*  CARD INPUT SPECIFICATIONS - CARD 1                                */
/*    COL 1 - 3     'AIM' OR 'ARM'                                    */
/*        4 - 6     LINE NO. OF FIRST PRINT LINE - INDX,SEASON,YR,PAGE*/
/*        7 -10     FIRST PAGE NO.                                    */
/*        11-13     LENGTH OF PAGES                                   */
/*        14-16     FIRST PAGE RIGHT OR LEFT - 1=LEFT  2=RIGHT        */
/*        17-19     WIDTH OF MARGIN 1                                 */
/*        20-22     WIDTH OF MARGIN 2 - 0 IF NO MARGIN                */
/*        23-25     WIDTH OF MARGIN 3 - 0 IF NO MARGIN                */
/*        26-28     WIDTH OF COLUMN - FROM 'V' IN VT NO TO 'E' IN PAGE*/
/*        29-31     NO. OF COLUMNS                                    */
/*        32-34     NO. OF SPACES BETWEEN VT NO AND ED NO             */
/*        35-37     LINE NO. OF LINE PRINTING 'PAGE' AS COLUMN HEADING*/
/*        38-40     LINE NO. OF FIRST INDEX DETAIL LINE - VT NO ED NO */
/*        41-43     LINE NO. ON TITLE PAGE OF LINE PRINTING 'PAGE'    */
/*        44-46     LINE NO. ON TITLE PAGE OF FIRST DETAIL LINE       */
/*                                                                    */
/*  CARD INPUT SPECIFICATIONS - CARD 2                                */
/*    COL 1 - 3     LENGTH OF TITLE                                   */
/*        4 - 6     LINE NO. OF TITLE TO BE PRINTED                   */
/*        7 -80     TITLE ITSELF                                      */
/*    ALLOCATE THE MATRICES USED TO STORE THE INFO TO BE PRINTED      */
/*    READ THE SORTOUT FILE                                           */
/*    EXECUTE THE ROUTINES TO ACCUMULATE THE DATA IN PRINTABLE FORMAT */
/*      MATRIX LINER - ACCUMULATE 1 PRINT RECORD (IE 1 VT NO REC)     */
/*      MATRIX BUILD - ACCUMULATE AN ENTIRE PAGE IN PRINTABLE FORMAT  */
/*      THE RECORDS ARE STORED IN BUILD IN SEQUENCE BY VT NO BY       */
/*        PRINT COLUMN                                                */
/*    PRINT AN ENTIRE PAGE FROM MATRIX BUILD WHEN BUILD IS FILLED     */
/*    RE INITIALIZE MATRIX BUILD AND BEGIN ACCUMULATING THE NEXT      */
/*      PAGE IN MATRIX BUILD                                          */
/*    'HANG IT UP' WHEN ALL RECORDS HAVE BEEN PROCESSED + PRINTED     */
/*    ERRONEOUS OUTPUT OR PROGRAM BLOWUPS - MOST LIKELY WILL BE DUE   */
/*      TO INCORRECT PRINT CONTROL CARDS WHICH ARE DEVISED BY THE     */
/*      USER - CHECK THESE CARDS FIRST.                               */
/*                                                                    */
    DECLARE
           ANNUAL   FILE OUTPUT RECORD,
           MASTER      FILE INPUT RECORD,
           SORTIN      FILE RECORD,
           SORTOUT     FILE RECORD,
           A_A                     CHAR (3),
           ANN                     CHAR (18),
           ANNUAL_REC              CHAR (24),
           BEG                     FIXED BINARY (31,0) INITIAL (0),
BLANK                      CHAR (1) INITIAL (' '),
BLD_CODE                   CHAR (1) INITIAL (' '),
BT                         CHAR (1) INITIAL ('<'),
BUILD   (NOCOL,PGE_LEN)    CONTROLLED  CHAR (COL),
BUILDIT                    ENTRY,
COL                        FIXED BINARY (31,0),
COL_L                      FIXED BINARY (31,0) INITIAL (0),
COLCTR                     FIXED BINARY (31,0) INITIAL (1),
DUM                        CHAR (35),
ED_NO                      CHAR (20),
EDIT_ISSUE                 CHAR (2),
EDIT_VT  (6)               CHAR (1),
ENDTIT                     CHAR (3) INITIAL ('   '),
ET                         CHAR (1) INITIAL ('>'),
FIRST_PAGE                 CHAR (1) INITIAL ('1'),
HEAD                       CHAR (74),
HINE                       CHAR (118),
I                          FIXED BINARY (31,0) INITIAL (0),
IHESRTA  ENTRY(CHAR(36),CHAR(33),FIXED BINARY(31,0),FIXED
           BINARY(15,0)),
INDX                       CHAR (6) VARYING,
ISKIP                      FIXED BINARY (31,0),
ISSUE                      CHAR (3),
J                          FIXED BINARY (31,0) INITIAL (0),
K                          FIXED BINARY (31,0),
L                          FIXED BINARY (31,0),
LEN_H                      FIXED BINARY (31,0),
LEN_TITLE                  FIXED BINARY (31,0),
LENTH (10)                 DECIMAL FIXED (6,0),
LIN                        CONTROLLED  CHAR (TIT_L),
LINE                       CHAR (118) INITIAL ((118) ' '),
LINE_DETAIL                FIXED BINARY (31,0),
LINE_H                     FIXED BINARY (31,0),
LINE_INDX                  FIXED BINARY (31,0),
LINE_L                     FIXED BINARY (31,0),
LINE_PAGE                  FIXED BINARY (31,0),
LINEH                      CHAR (118) INITIAL((118) ' '),
LINER    (TIT_MAX)         CONTROLLED  CHAR (COL_L),
LINERF                     CONTROLLED  CHAR (COL),
LINESLEFT                  FIXED BINARY (31,0) INITIAL (0),
LNECTR                     FIXED BINARY (31,0) INITIAL (1),
M                          FIXED BINARY (31,0),
MAST_REC                   CHAR (3736) VARYING,
MG1                        FIXED BINARY (31,0),
MG2                        FIXED BINARY (31,0),
MG3                        FIXED BINARY (31,0),
N                          FIXED BINARY (31,0),
NOCOL                      FIXED BINARY (31,0),
PAGE_NO                    CHAR (4),
PGE_LEN                    FIXED BINARY (31,0),
PGE_NO                     FIXED BINARY (31,0),
PGE_RL                     FIXED BINARY (31,0),
PRINTIT                    ENTRY,
RETCDE                     FIXED BINARY (15,0),
RFIELD                     CHAR (33),
SFIELD                     CHAR (36),
            SPACES                  CHAR (118) INITIAL ((118) ' '),
            SUBLINER                FIXED BINARY (31,0) INITIAL (1),
            SUM1                    BINARY FIXED (31,0),
            TIT (3)                 CHAR (396) VARYING,
            TIT_L                   FIXED BINARY (31,0) INITIAL (0),
            TIT_MAX                 FIXED BINARY (31,0) INITIAL (0),
            TITLE                   CHAR (396) VARYING,
            TITLE_LEN               DECIMAL FIXED (6,0),
            TOP_PAGE                ENTRY,
            TP_LIN_DET              FIXED BINARY (31,0),
            TP_LIN_PGE              FIXED BINARY (31,0),
            UNPACK                  CHAR (11),
            VARYER                  DECIMAL FIXED (6,0),
            VT_NO                   CHAR (10) VARYING,
            YEAR1                   CHAR (2),
            VT_ED                   FIXED BINARY (31,0),
            YEAR2                   CHAR (2);
     BEGIN:
          OPEN FILE (MASTER),
               FILE (SORTIN) OUTPUT;
     READ_MAST:
            /**********************************************************/
            READ FILE (MASTER) INTO (MAST_REC);
              ON ENDFILE (MASTER) GO TO SORTIT;
            /**********************************************************/
            N = INDEX(MAST_REC,'||');
            SUM1 = 0;
            GET STRING (MAST_REC)  EDIT (VARYER) (X(N+1),F(4,0));
            IF VARYER = 0 THEN GET STRING (MAST_REC) EDIT (TITLE_LEN)
                                                   (X(N+5),F(4,0));
              ELSE DO;
                GET STRING (MAST_REC) EDIT ((LENTH(I) DO I=1 TO VARYER),
                            TITLE_LEN) (X(N+5),(VARYER+1) F(4,0));
     THRU_AUTHOR:
                  DO I = 1 TO VARYER;
                  SUM1 = SUM1 + LENTH(I);
                  END THRU_AUTHOR;
              END;
            /**********************************************************/
 /*  EXTRACT DATA FROM MASTER FILE - VT NO  ED NO  ISSUE  YEAR         */
 /*                                  PAGE NO  TITLE                    */
            GET STRING (MAST_REC) EDIT(VT_NO,ED_NO,ISSUE,YEAR1,YEAR2,
                    PAGE_NO,TITLE)
            (A(10),X(17),A(20),A(3),A(2),A(2),A(4),X(SUM1),A(TITLE_LEN));
            /**********************************************************/
            IF TITLE_LEN = 0 THEN DO;
              PUT EDIT ('RECORD WITH A TITLE LENGTH OF ZERO ',VT_NO)
              (SKIP(1),2 A);
              GO TO READ_MAST;
              END;
            I = 0;
 /*  ROUTINE TO DETERMINE THE DOCUMENT TITLE BY BREAKING TITLE BY      */
 /*     BEGIN AND  END TITLE CHARACTERS                                */
    TIT_BREAK:
            N = INDEX(TITLE,BT);
            IF N = 0 THEN GO TO CAT;
```

```
           M = INDEX(TITLE,ET);
           SUBSTR (TITLE,M,1) = SUBSTR (TITLE,M+1,1);
           L = M - N;
           I = I + 1;
           TIT(I) = SUBSTR(TITLE,N+1,L);
           TITLE = SUBSTR(TITLE,M+1);
           GO TO TIT_BREAK;
    CAT:   TITLE = '';
    TIT_BREAKER:
           DO J = 1 TO I;
           IF J = 1 THEN TITLE = TITLE || TIT(1);
             ELSE         TITLE = TITLE || BLANK || TIT(J);
           END TIT_BREAKER;
           TITLE_LEN = LENGTH(TITLE);
 /*  WRITE RECORDS TO BE SORTED BY THE VT NO                           */
    FOR_SORT:
           LEN_TITLE = TITLE_LEN;
           /***********************************************************/
           PUT STRING (MAST_REC) EDIT (VT_NO,ED_NO,PAGE_NO,LEN_TITLE,
                       TITLE,ISSUE,YEAR1,YEAR2)
                      (A(10),A(20),A(4),F(4),A(TITLE_LEN),A(3),2 A(2));
           WRITE FILE (SORTIN) FROM (MAST_REC);
           PUT EDIT ('UNSORTED EXTRACT ',MAST_REC) (SKIP(2),2 A);
           /***********************************************************/
           GO TO READ_MAST;
           /***********************************************************/
           /***********************************************************/
    SORTIT:
           CLOSE FILE (SORTIN) , FILE (MASTER);
           SFIELD = ' SORT FIELDS=(5,10,CH,A),SIZE=E900 ';
           RFIELD = ' RECORD TYPE=V,LENGTH=(450,,,29) ';
           CALL IHESRTA(SFIELD,RFIELD,45056,RETCDE);
           IF RETCDE = 16 THEN PUT EDIT ('RETURN CODE OF 16') (A(15));
           /***********************************************************/
           /***********************************************************/
           OPEN FILE (SORTOUT);
           ON ENDFILE (SORTOUT) BEGIN;
             PGE_NO = PGE_NO + 1;
             CALL TOP_PAGE;
             CALL PRINTIT;
             GO TO PRINT_FIN;
             END;
 /*  READ THE 2 PRINT CONTROL CARDS                                    */
    SETUP:
           GET FILE (SYSIN) EDIT (A_A,LINE_INDX) (A(3),F(3));
           GET FILE (SYSIN) EDIT (PGE_NO,PGE_LEN,PGE_RL,MG1,MG2,
             MG3,COL,NOCOL,VT_ED,LINE_PAGE,LINE_DETAIL,TP_LIN_PGE,
             TP_LIN_DET) (F(4),12(F(3)));
           GET FILE (SYSIN) EDIT (DUM) (A(34));
           GET FILE (SYSIN)  EDIT (LEN_H,LINE_H,HEAD) (2 F(3),A(74));
           PGE_NO = PGE_NO - 1;
           COL_L = COL - 3;
           TIT_L = COL - 5;
           TIT_MAX = (396/TIT_L) + 4;
           LINE_L = (MG1 + MG2 + MG3 + (NOCOL * COL));
           EDIT_ISSUE = SUBSTR(ISSUE,2,2);
```

```
          IF EDIT_ISSUE = 'W ' THEN INDX = 'WINTER';
          IF EDIT_ISSUE = 'F ' THEN INDX = 'FALL';
          IF EDIT_ISSUE = 'SP' THEN INDX = 'SPRING';
          IF EDIT_ISSUE = 'SU' THEN INDX = 'SUMMER';
          OPEN FILE (INDIX) OUTPUT STREAM PRINT
            PAGESIZE(PGE_LEN);
          PGE_LEN = PGE_LEN - (LINE_DETAIL - 1);
          ALLOCATE BUILD;
          ALLOCATE LIN;
          ALLOCATE LINER;
          ALLOCATE LINERF;
          CALL CLEAN_BUILD;
          PGE_LEN = PGE_LEN - (TP_LIN_DET - LINE_DETAIL);
          GO TO READ_SORTOUT;
/*  ROUTINE TO PRINT THE HEADINGS AT THE TOP OF PAGE                         */
  TOP_PAGE:  PROCEDURE;
          IF PGE_RL = 1 THEN
            PUT FILE (INDIX) EDIT (PGE_NO,A_A,'/',INDX,' ',YEAR2)
           (PAGE,LINE(LINE_INDX),X(MG1),F(4),X(LINE_L-(MG1+4+7+
            LENGTH(INDX))),A(3),A(1),A,A(1),A(2));
            ELSE
          PUT FILE (INDIX) EDIT (A_A,'/',INDX,' ',YEAR2,PGE_NO)
             (PAGE,LINE(LINE_INDX),X(MG1),A(3),A(1),A,A(1),A(2),
            X(LINE_L-(MG1+4+7+LENGTH(INDX))),F(4));
          IF FIRST_PAGE = '0' THEN
          PUT FILE (INDIX) EDIT (HEAD) (LINE(LINE_H),X((LINE_L/2)-
            (LEN_H/2)),A(21));
          LINE = SPACES;
          IF NOCOL < 3 THEN GO TO T2;
          SUBSTR(LINE,MG1+MG2+MG3+(3*COL)-3,4) = 'PAGE';
    T2:   IF NOCOL< 2 THEN GO TO T1;
          SUBSTR(LINE,MG1+MG2+(2*COL)-3,4) = 'PAGE';
    T1: . SUBSTR(LINE,MG1+(COL-3),4) = 'PAGE';
          HINE = LINE;
          IF FIRST_PAGE = '0' THEN
          PUT FILE (INDIX) EDIT (HINE) (LINE(LINE_PAGE),A(118));
           ELSE PUT FILE (INDIX) EDIT (HINE) (LINE(TP_LIN_PGE),A(118));
          IF FIRST_PAGE = '0' THEN
            ISKIP = (LINE_DETAIL - (LINE_PAGE + 1));
            ELSE ISKIP = (TP_LIN_DET - (TP_LIN_PGE + 1));
          PUT FILE (INDIX) EDIT (SPACES) (SKIP(ISKIP),A(118));
          IF PGE_RL = 1 THEN PGE_RL = 0;  ELSE PGE_RL = 1;
          RETURN;
          END TOP_PAGE;
/*  READ THE SORTED FILE                                                     */
  READ_SORTOUT:
          /*****************************************************************/
          READ FILE (SORTOUT) INTO (MAST_REC);
          GET STRING (MAST_REC) EDIT (VT_NO,ED_NO,PAGE_NO,LEN_TITLE,
                      TITLE,ISSUE,YEAR1,YEAR2)
                  (A(10),A(20),A(4),F(4),A(LEN_TITLE),A(3),2 A(2));
    IF VT_NO='VT 010 625' THEN GO TO READ_SORTOUT;
          /*****************************************************************/
          CALL ANNUAL_INDEX;
/*  ROUTINE TO TEMPORARILY STORE THE DOCUMENT TITLE IN PRINTABLE FORM*/
/*    IN THE MATRIX LINER                                                    */
```

```
          LINE = SPACES;
          PUT STRING (LINE) EDIT (VT_NO,ED_NO,PAGE_NO)
            (A(10),X(VT_ED),A(20),X(COL-(34+VT_ED)),A(4));
          SUBLINER = SUBLINER + 1;  LNECTR = LNECTR + 1;
          BEG = 1;
   BRK_TIT:
          IF LEN_TITLE < TIT_L | LEN_TITLE = TIT_L THEN GO TO END_VT;
          LIN = SUBSTR(TITLE,BEG,TIT_L);
          IF SUBSTR(TITLE,(BEG+TIT_L),1) = ' ' THEN DO;
            LINER(SUBLINER) = ('  ' || LIN);
            BEG = BEG + TIT_L + 1;
            LEN_TITLE = LEN_TITLE - (TIT_L + 1);
            SUBLINER = SUBLINER + 1;  LNECTR = LNECTR + 1;
            GO TO BRK_TIT;
            END;
          DO J = TIT_L TO 1 BY -1 WHILE (SUBSTR(LIN,J,1) ¬= ' '); END;
          LINER(SUBLINER) = ('  ' || SUBSTR(LIN,1,J-1));
          BEG = BEG + J;
          LEN_TITLE = LEN_TITLE - J;
          SUBLINER = SUBLINER + 1;  LNECTR = LNECTR + 1;
          GO TO BRK_TIT;
   END_VT:
          LINER(SUBLINER) = ('  ' || SUBSTR(TITLE,BEG,LEN_TITLE));
          GO TO CHEK_BOT;
/*  ROUTINE TO HANDLE BOTTOM OF PAGE IN THE MATRIX BUILD                */
   CHEK_BOT:
          LINESLEFT = PGE_LEN - LNECTR;
          IF LINESLEFT > 2 THEN DO;
            BLD_CODE = '1';
            CALL BUILDIT;
            BLD_CODE = ' ';
            GO TO READ_SORTOUT;
            END;
          IF LINESLEFT >= 0 THEN GO TO FINISH_COL;
            IF COLCTR = NOCOL THEN DO;
              LINEH = LINE;
              PGE_NO = PGE_NO + 1;
              CALL TOP_PAGE;
              LINE = LINEH;
              CALL PRINTIT;
              LNECTR = SUBLINER;
              BLD_CODE = '1';
              CALL BUILDIT;
              BLD_CODE = ' ';
              GO TO READ_SORTOUT;
              END;
            LNECTR = SUBLINER;
            COLCTR = COLCTR + 1;
            BLD_CODE = '1';
            CALL BUILDIT;
            BLD_CODE = ' ';
            GO TO READ_SORTOUT;
    FINISH_COL:
          IF COLCTR = NOCOL THEN DO;
            CALL BUILDIT;
            PGE_NO = PGE_NO + 1;
            CALL TOP_PAGE;
            CALL PRINTIT;
            GO TO READ_SORTOUT;
            END;
          CALL BUILDIT;
          COLCTR = COLCTR + 1;
          LNECTR = 1;
          GO TO READ_SORTOUT;
          /************************************************************/
/*  ROUTINE TO BUILD AN ENTIRE PRINT PAGE IN THE MATRIX BUILD         */
   BUILDIT:    PROCEDURE;
          J = 2;
          LNECTR = LNECTR + 1;
          I = LNECTR - SUBLINER;
          BUILD(COLCTR,I) = LINE;
          SUBLINER = SUBLINER - 1;
          DO I = SUBLINER  TO 1 BY -1;
          BUILD(COLCTR,LNECTR-I) = LINER (J);
          J = J + 1;
          END;
          IF BLD_CODE = '1' THEN DO;
            BUILD(COLCTR,LNECTR) = SPACES;
            LNECTR = LNECTR + 1;
            END;
          DO I = 1 TO TIT_MAX;
          LINER (I) = SPACES;
          END;
          SUBLINER = 1;
          RETURN;
          END BUILDIT;
          /************************************************************/
          /************************************************************/
/*  ROUTINE TO PRINT AN ENTIRE PAGE FROM THE  MATRIX BUILD            */
   PRINTIT:        PROCEDURE;
          DO I = 1 TO PGE_LEN;
          IF NOCOL = 1 THEN PUT FILE (INDIX) EDIT (BUILD(1,I))
            (SKIP(1),X(MG1),A(COL));
          IF NOCOL = 2 THEN PUT FILE (INDIX) EDIT (BUILD(1,I),
            BUILD(2,I)) (SKIP(1),X(MG1),A(COL),X(MG2),A(COL));
          IF NOCOL = 3 THEN PUT FILE (INDIX) EDIT (BUILD(1,I),
            BUILD(2,I),BUILD(3,I)) (SKIP(1),X(MG1),A(COL),X(MG2),
            A(COL),X(MG3),A(COL));
          END;
          CALL CLEAN_BUILD;
          IF FIRST_PAGE = '1' THEN DO;
            FIRST_PAGE = '0';
            PGE_LEN = PGE_LEN + (TP_LIN_DET - LINE_DETAIL);
            END;
          COLCTR = 1;
          LNECTR = 1;
          RETURN;
          END PRINTIT;
          /************************************************************/
   CLEAN_BUILD:   PROCEDURE;
          DO I = 1 TO PGE_LEN;
          IF NOCOL >= 1 THEN BUILD(1,I) = SPACES;
          IF NOCOL >= 2 THEN BUILD(2,I) = SPACES;
          IF NOCOL = 3 THEN BUILD(3,I) = SPACES;
            END;
            RETURN;
          END CLEAN_BUILD;
          /**********************************************************/
/*  ROUTINE TO WRITE OUT THE ANNUAL FILE - VT NO  PAGE NO  ISSUE      */
/*                                        YEAR  -  18 BYTE RECORDS  */
   ANNUAL_INDEX:  PROCEDURE;
          K = 0;
   EDIT_VTEE:
          DO J = 1 TO 10;
          IF SUBSTR(VT_NO,J,1) = 'V' |
              SUBSTR(VT_NO,J,1) = 'T' |
              SUBSTR(VT_NO,J,1) = ' '
              THEN GO TO E_L;
          ELSE DO;
            K = K + 1;
            EDIT_VT (K) = SUBSTR(VT_NO,J,1);
            END;
   E_L:   END EDIT_VTEE;
          /**********************************************************/
          PUT STRING (ANN) EDIT (EDIT_VT,PAGE_NO,ISSUE,YEAR2,'   ')
            (6 A(1),A(4),A(3),A(2),A(3));
          WRITE FILE (ANNUAL) FROM (ANN);
          PUT EDIT ('ANNUAL FILE ',ANN) (SKIP(1),2 A);
          /**********************************************************/
          RETURN;
          END ANNUAL_INDEX;
   PRINT_FIN:
          CLOSE FILE (SORTOUT) , FILE (INDIX);
          END;
```

CONVERSION OF DOCUMENT NUMBER

```
/**********************************************************************/00000020
/*                                                                    */00000030
/*           Q U A R T E R L Y   A N D   A N N U A L                  */00000040
/*   C O N V E R S I O N   O F   D O C U M E N T     N U M B E R      */00000050
/*                      I N D E X                                     */00000060
/*                                                                    */00000070
/*                                                                    */00000080
/*  PURPOSE:  THIS PROGRAM CAN GENERATE A QUARTERLY OR AN ANNUAL      */00000090
/*            CONVERSION OF DOCUMENT NUMBER INDEX.  IF WE ARE         */00000100
/*            INTERESTED IN A QUARTERLY INDEX, WE INPUT TO THE        */00000110
/*            PROGRAM THE AIM OR ARM CONVERSION TAPE WITH ITS ONE     */00000120
/*            LARGE DATA SET.  FOR AN ANNUAL PRINTOUT, WE USE BOTH    */00000130
/*            AIM AND ARM CONVERSION TAPE DATA SETS (THEY MAY BE ON  */00000140
/*            ONE OR TWO TAPES).  FOR EITHER INDEX, THE VOCATIONAL    */00000150
/*            CENTER SPECIFIES THE PAGE DIMENSIONS AND AN INDEX IS    */00000160
/*            PRODUCED FROM THE TAPE DATA.  THERE IS NO SHORTENING    */00000170
/*            OF RECORDS.  THE QUARTERLY TAPES ARE SAVED AS IS FOR    */00000180
/*            THE ANNUAL INDEX.                                       */00000190
/*                                                                    */00000200
/*  DEFINITION OF VARIABLES:                                          */00000210
/*    UNLESS OTHERWISE SPECIFIED, VARIABLES ARE THE SAME FOR EITHER   */00000220
/*    INDEX.                                                          */00000230
/*      1. COL -- NUMBER OF COLUMNS ON AN INDEX PAGE.                 */00000240
/*     2. HEAD -- HEADING IN A TOP CORNER OF EVERY PAGE WHICH TELLS  */00000250
/*                THE ISSUE, SEASON, AND YEAR FOR A QUARTERLY         */00000260
/*                PRINTOUT (EX: FALL/AIM 69) OR IT TELLS ISSUE AND    */00000270
/*                YEAR FOR AN ANNUAL PRINTOUT (EX: AIM/ARM 69-70).    */00000280
/*     3. INDX -- INDICATES WHETHER AN ANNUAL OR QUARTERLY INDEX IS  */00000290
/*                TO BE PRODUCED.  'Q' MEANS QUARTERLY; 'A' MEANS     */00000300
/*                ANNUAL.                                             */00000310
/*     4. LINEH1 -- LINE ON THE TITLE PAGE ON WHICH THE INDEX PAGE    */00000320
/*                  NUMBER AND -HEAD- ARE FOUND, AT OPPOSITE ENDS.    */00000330
/*     5. LINEH2 -- SAME AS -LINEH1-BUT FOR ALL PAGES EXCEPT THE      */00000340
/*                  TITLE PAGE.                                       */00000350
/*     6. LINES1 -- LINE ON THE TITLE PAGE OF THE QUARTERLY INDEX     */00000360
/*                  ON WHICH THE SUB-TITLE IS FOUND; LINE ON THE      */00000370
/*                  TITLE PAGE OF THE ANNUAL INDEX RIGHT ABOVE TH     */00000380
/*                  START OF THE ACTUAL INDEX PRINTOUT.               */00000390
/*     7. LINES2 -- SAME AS -LINES1- BUT FOR ALL PAGES EXCEPT THE     */00000400
/*                  TITLE PAGE.                                       */00000410
/*     8. LINET -- LINE ON ALL PAGES EXCEPT THE TITLE PAGE ON WHICH  */00000420
/*     9. LH -- LENGTH OF VARIABLE -HEAD-.                            */00000430
/*    11. LT -- LENGTH OF INDEX TITLE.                                */00000440
/*    12. MARG -- LEFT-HAND MARGIN OF THE INDEX PAGE.                 */00000450
/*    13. MG -- ARRAY OF COLUMN MARGINS, EACH MEASURED FROM COLUMN   */00000460
/*                 THE TITLE APPEARS.  THERE IS NO TITLE ON THE       */00000470
/*                 TITLE PAGE.                                        */00000480
/*              THE ANNUAL INDEX.                                     */00000490
/*    10. LS -- LENGTH OF SUB-TITLE FOR THE QUARTERLY INDEX; N/A FOR */00000500
/*              1.                                                    */00000510
/*    14. NBH -- NUMBER OF BLOCKS OF PRINT ON THE TITLE PAGE.         */00000520
/*    15. NBR -- NUMBER OF BLOCKS OF PRINT ON ALL OTHER PAGES.        */00000530
/*    16. NPB -- NUMBER OF LINES OF PRINT PER BLOCK.                  */00000540
/*    17. PAGE -- PAGE NUMBER OF INDEX PAGES.                         */00000550
/*    18. QUE -- INDICATES WHETHER THE TITLE PAGE WILL BE ON THE      */00000560
/*               LEFT ('L') OR RIGHT ('R') SIDE OF THE PUBLICATION.   */00000570
/*    19. SIZE -- NUMBER OF LINES OF PRINT PER PAGE INCLUDING ALL     */00000580
/*                HEADINGS.                                           */00000590
/*    20. SPACE -- USED IN QUARTERLY INDEX ONLY.  NUMBER OF LINES     */00000600
/*                 SKIPPED BETWEEN THE LAST LINE OF PRINT ON THE      */00000610
/*                 TITLE PAGE AND THE FOOTNOTE (SET AT 3 NOW).        */00000620
/*    21. SUB -- SUB-TITLE ON EVERY PAGE OF THE QUARTERLY INDEX.      */00000630
/*     22. SZ -- RIGHT-HAND MARGIN OF THE INDEX PAGE.                          */00000640
/*     23. TITL -- TITLE OF THE INDEX.                                         */00000650
/*     24. WIDTH -- COLUMN WIDTH.                                              */00000660
/*     25. COL# -- A TWO-DIMENSIONAL ARRAY REPRESENTING A PAGE OF              */00000670
/*                 OUTPUT UNDER THE PAGE TITLE.  THE FIRST SUBSCRIPT           */00000680
/*                 REFERENCES A COLUMN NUMBER AND THE SECOND SUBSCRIPT*/00000690
/*                 REFERENCES A LINE IN THE COLUMN.                            */00000700
/*     26. ED -- ACCESSION NUMBER.                                             */00000710
/*     27. IHESRTA -- CALL TO SORT PROGRAM.                                    */00000720
/*     28. IMP -- SIGNIFICANT TO QUARTERLY PROGRAM ONLY.  IMP = 1              */00000730
/*                FOR THE VT TO ED OR MP NUMBER SECTION; IMP = 2 FOR           */00000740
/*                ED OR MP NUMBER TO VT NUMBER SECTION.                        */00000750
/*     29. IND -- COLUMN NUMBER BEING FILLED.                                  */00000760
/*     30. NN -- LINE NUMBER BEING FILLED IN THAT COLUMN.                      */00000770
/*     31. RETCDE -- RETURN CODE RETURNED BY SORT TO INDICATE SUCCESS */00000780
/*                   OR FAILURE.                                               */00000790
/*     32. RFIELD -- DEFINES THE TYPE OF RECORDS BEING SORTED FOR              */00000800
/*                   SORT.                                                     */00000810
/*     33. SFIELD -- DEFINES THE PARTS OF THE RECORDS TO BE SORTED.            */00000820
/*     34. SKPH -- MARGIN FOR THE HEADING -HEAD- ON ITS PROPER LINE.           */00000830
/*     35. SKPP -- MARGIN FOR THE PAGE NUMBER OF THE INDEX ON ITS              */00000840
/*                 PROPER LINE.                                                */00000850
/*     36. SORTIN -- CONVERSION TAPE DATA SET(S).                              */00000860
/*     37. SORTED -- SORTED DISK SUBSET FILE.                                  */00000870
/*     38. STR -- STRING USED IN INPUTING AND OUTPUTING OF FILE                */00000880
/*                RECORDS.                                                     */00000890
/*     39. UNDER -- USED IN QUARTERLY INDEX ONLY.  THE LINE SEPARATING*/00000900
/*                  THE INDEX AND FOOTNOTE ON THE TITLE PAGE.                  */00000910
/*     40. VT -- DOCUMENT NUMBER FROM MASTER.                                  */00000920
/*                                                                             */00000930
/*  INPUT TO PROGRAM:                                                          */00000940
/*    -CONVERSION TAPE-                                                        */00000950
/*      EITHER THE AIM OR ARM DATA SET IS USED FOR THE QUARTERLY               */00000960
/*      INDEX, OR BOTH DATA SETS ARE USED FOR AN ANNUAL INDEX.  IF             */00000970
/*      THE DATA SETS ARE ON TWO SEPARATE TAPES, CONCATENATE THE               */00000980
/*      TWO DD CARDS.                                                          */00000990
/*    -SYSIN-, INPUT ON CARDS                                                  */00001000
/*      THE CARD INPUT IS ONE CONTINUOUS STREAM OF DATA.  THE VALUES */00001010
/*      OF THE FOLLOWING VARIABLES ARE FOUND ON THE CARDS IN THE               */00001020
/*      SAME ORDER AS THEY ARE LISTED BELOW AND IN THE FORMAT LISTED */00001030
/*      BELOW, TOO.                                                            */00001040
/*              -PAGE-                  F(4,0)                                 */00001050
/*              -SIZE-                  F(4,0)                                 */00001060
/*              -WIDTH-                 F(4,0)                                 */00001070
/*              -MARG-                  F(4,0)                                 */00001080
/*              -SZ-                    F(4,0)                                 */00001090
/*              -LINEH1-                F(4,0)                                 */00001100
/*              -LINES1-                F(4,0)                                 */00001110
/*              -LINEH2-                F(4,0)                                 */00001120
/*              -LINET-                 F(4,0)                                 */00001130
/*              -LINES2-                F(4,0)                                 */00001140
/*              -SPACE-                 F(4,0)                                 */00001150
/*              -COL-                   F(4,0)                                 */00001160
/*              -MG- ARRAY              COL * F(4,0)                           */00001170
/*              -NPB-                   F(4,0)                                 */00001180
/*              -NBH-                   F(4,0)                                 */00001190
/*              -NBR-                   F(4,0)                                 */00001200
/*              -LT-                    F(4,0)                                 */00001210
/*              -TITL-                  A(LT)                                  */00001220
/*              -LH-                    F(4,0)                                 */00001230
/*              -HEAD-                  A(LH)                                  */00001240
/*              -INDX-                  A(1)                                   */00001250
/*            -QUE-                 A(1)                                 */
/*         FOR QUARTERLY INDEX ONLY, READ IN:                            */
/*            -LS- (FOR SECTION 1) F(4,0)                                */
/*            -SUB-                 A(LS)                                */
/*            -LS- (FOR SECTION 2) F(4,0)                                */
/*            -SUB-                 A(LS)                                */
/*      THEY ARE DEFINED IN THE PRECEEDING SECTION AND THEIR VALUES      */
/*      CAN BE FOUND IN TYPED WRITE-UP OR CAN BE SUPPLIED BY THE         */
/*      VOCATIONAL CENTER.                                               */
/*                                                                       */
/*  SPECIAL NOTE:                                                        */
/*    UNTIL THE VOCATIONAL CENTER APPROVES AN INDEX PRINTOUT, USE        */
/*    THE PRINTER AS THE OUTPUT FILE.  THEN AFTER APPROVAL FROM THEM,*/
/*    CHANGE THE OUTPUT FILE TO TAPE.  LIST THE TAPE ON THE 360/50       */
/*    WITH THE TN CHAIN FOR UPPER AND LOWER CASE LETTERS.                */
/*                                                                       */
/***********************************************************************/

PRNTCV: PROCEDURE OPTIONS (MAIN);
        DECLARE
          COL                 FIXED BINARY (31,0),
          HEAD                CHARACTER (13) VARYING,
          I                   FIXED BINARY (31,0),
          INDX                CHARACTER (1),
          LINEH1              FIXED BINARY (31,0),
          LINEH2              FIXED BINARY (31,0),
          LINES1              FIXED BINARY (31,0),
          LINES2              FIXED BINARY (31,0),
          LINET               FIXED BINARY (31,0),
          LH                  FIXED BINARY (31,0),
          LS                  FIXED BINARY (31,0),
          LT                  FIXED BINARY (31,0),
          M                   FIXED BINARY (31,0),
          MARG                FIXED BINARY (31,0),
          MG(5)               FIXED BINARY (31,0),
          NBH                 FIXED BINARY (31,0),
          NBR                 FIXED BINARY (31,0),
          NPB                 FIXED BINARY (31,0),
          PAGE                FIXED BINARY (31,0),
          QUE                 CHARACTER (1),
          SIZE                FIXED BINARY (31,0),
          SPACE               FIXED BINARY (31,0),
          SUB                 CHARACTER (48),
          SZ                  FIXED BINARY (31,0),
          TITL                CHARACTER (35),
          WIDTH               FIXED BINARY (31,0),
         ;


 /*  READ IN THE PAGE SPECIFICATIONS FOR THE INDEX PRINTOUT.  ONLY THE
     QUARTERLY INDEX USES THE SUB-TITLE.                                */


         GET FILE (SYSIN) EDIT (PAGE,SIZE,WIDTH,MARG,SZ,LINEH1,LINES1,
           LINEH2,LINET,LINES2,SPACE,COL,(MG(I) DO I = 1 TO COL),NPB,
           NBH,NBR,LT,TITL,LH,HEAD,INDX,QUE)
           (16 F(4),(COL) F(4),A(LT),F(4),A(LH),2 A(1));
         IF INDX = 'Q' THEN GET FILE (SYSIN) EDIT (LS,SUB) (F(4),A(LS));
         IF INDX = 'Q' THEN M = LINES2 + 1;
                       ELSE M = LINES2;

 /*  PRINTING OF THE INDEX IS ACCOMPLISHED IN THIS BLOCK.            */00001880
                                                                        00001890
                                                                        00001900
WORK:    BEGIN;                                                         00001910
           DECLARE                                                      00001920
           COL#(COL,SIZE-M)   CHARACTER (WIDTH),                        00001930
           ED                 CHARACTER (10),                           00001940
           IHESRTA            ENTRY(CHARACTER (45), CHARACTER (27),     00001950
                              FIXED BINARY (31,0), FIXED BINARY (15,0)), 00001960
           IMP                FIXED BINARY (31,0),                      00001970
           IND                FIXED BINARY (31,0),                      00001980
           J                  FIXED BINARY (31,0),                      00001990
           K                  FIXED BINARY (31,0),                      00002000
           L                  FIXED BINARY (31,0),                      00002010
           NN                 FIXED BINARY (31,0),                      00002020
           RETCDE             FIXED BINARY (15,0),                      00002030
           RFIELD             CHARACTER (27),                           00002040
            SFIELD            CHARACTER (45),                           00002050
           SKPH               FIXED BINARY (31,0),                      00002060
           SKPP               FIXED BINARY (31,0),                      00002070
           SORTIN             FILE SEQUENTIAL RECORD,                   00002080
           SORTOUT            FILE SEQUENTIAL RECORD,                   00002090
           STR                CHARACTER (20),                           00002100
           UNDER              CHARACTER (50) INITIAL ((50)'_'),         00002110
           VT                 CHARACTER (10),                           00002120
                                                                        00002130
         ;                                                              00002140
                                                                        00002150
 /*  BRANCH HERE WHEN THE SORTED FILE HAS BEEN PRINTED COMPLETELY AND  00002160
     ALL ITS RECORDS READ.                                           */00002170
                                                                        00002180
                                                                        00002190
          ON ENDFILE (SORTOUT) BEGIN;                                   00002200
           CLOSE FILE (SORTOUT);                                        00002210
           IMP = IMP + 1;                                               00002220
           IF (NN = 1) & (IND = 1) THEN GO TO TEST;                     00002230
                                                                        00002240
                                                                        00002250
 /*  PRINT THE LAST PAGE OF THE INDEX THAT WAS BEING FILLED AND PREPARE00002260
     FOR A NEW TITLE ON THE NEXT PAGE.                               */00002270
                                                                        00002280
                                                                        00002290
           LOOP5: DO I = 1 TO L * NPB + L - 1;                          00002300
                  PUT FILE (OUTS) SKIP(1);                              00002310
           LOOP6: DO J = 1 TO COL;                                      00002320
                  PUT FILE (OUTS) EDIT (COL#(J,I))                      00002330
                    (SKIP(0),COLUMN(MG(J)),A(WIDTH));                   00002340
                  COL#(J,I) = ' ';                                      00002350
                  END LOOP6;                                            00002360
                  END LOOP5;                                            00002370
            IF SKPP = MARG  THEN DO;                                    00002380
                                 SKPP = SZ - 3;                         00002390
                                 SKPH = MARG;                           00002400
                                 END;                                   00002410
                            ELSE DO;                                    00002420
                                 SKPH = SZ - LH + 1;                    00002430
                                 SKPP = MARG;                           00002440
                                 END;                                   00002450
                                                                        00002460
                                                                        00002470
                          /* THIS IS THE FOOTNOTE THAT APPERS           00002480
                             ON THE TITLE PAGE OF THE QUARTERLY         00002490
                          INDEX.                                    */


              IF K=PAGE&INDX='O' THEN  DO;
               K=SIZE-LINENO(OUTS)-SPACE-7;
               PUT FILE(OUTS) SKIP(K);
               K=PAGE;
                PUT FILE (OUTS) EDIT (UNDER,'1','THIS INDEX IS CUMUL',
                'ATED ANNUALLY BEGINNING WITH THE FALL ISSUE OF ARM.',
                '2','ANY DOCUMENT WITH AN MP NUMBER IS AVAILABLE FROM'
                ,' EDRS BY USING THE','APPROPRIATE MP NUMBER.')
                (SKIP(SPACE),COLUMN(MG(1)-3),A(37),SKIP(1),COLUMN(MG
                (1)-1),A(1),SKIP(1),COLUMN(MG(1)),A(19),A(51),SKIP(1),
                COLUMN(MG(1)-1),A(1),SKIP(1),COLUMN(MG(1)),A(48),A(18)
                ,SKIP(1),COLUMN(MG(1)),A(22));
                                       END;
         PAGE = PAGE + 1;
/*   IF WE ARE DOING AN ANNUAL INDEX, THEN AFTER THE VT TO ED NUMBER
     SECTION IS FINISHED WE END THE PROGRAM.  IF SECTION 2 OF THE
     QUARTERLY INDEX IS FINISHED, WE END THE PROGRAM ALSO.            */


         TEST: IF IMP ¬= 2 THEN GO TO STOP;
         IF INDX = 'A' THEN GO TO STOP;


/*   TO PREPARE FOR SECTION 2 OF THE QUARTERLY INDEX, WE PUT A TITLE
     ON THE NEXT PAGE AND REVISE THE SORT CONTROL STATEMENTS.  ALSO, A
     NEW SUB-TITLE IS READ IN.  RETURN FOR FURTHER PROCESSING.        */


         SFIELD = ' SORT FIELDS=(11,2,CH,D,14,7,CH,A),SIZE=E600 ';
         GET FILE (SYSIN) EDIT (LS,SUB) (F(4),A(LS));
         PUT FILE (OUTS) EDIT (PAGE,HEAD,TITL,SUB,UNDER)
           (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
           A(LH),LINE(LINET),X((SZ-MARG+1-LT)/2+MARG-1),A(LT),LINE(
           LINES2),X((SZ-MARG+1-LS)/2+MARG-1),A(LS),SKIP(0),X((SZ-MARG
           +1-LS)/2+MARG-1),A(LS));
         PUT FILE (OUTS) SKIP(1);
         GO TO SORT;
         END;


/*   INITIALIZE VARIABLES FOR PRINTING.                               */


        OPEN FILE (OUTS) STREAM OUTPUT PRINT PAGESIZE (SIZE+4)
        ;
        IMP = 1;
        K = PAGE;
BLKOT:  DO I = 1 TO SIZE-M;
        DO J = 1 TO COL;
        COL#(J,I) = ' ';
        END BLKOT;
        IF QUE = 'L' THEN DO;
                          SKPP = MARG;
                          SKPH = SZ - LH + 1;
                          END;
                     ELSE DO;
                          SKPH = MARG;
                          SKPP = SZ - 3;
                          END;
        L = NBH;


/*  PUT A TITLE ON THE TITLE PAGE.  PUT A SUB-TITLE ON THE PAGE IF
    WE ARE DOING A QUARTERLY INDEX.                                   */


        PUT FILE (OUTS) EDIT (PAGE,HEAD)
          (PAGE,LINE(LINEH1),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
          A(LH));
        IF INDX = 'A' THEN DO;
                   PUT FILE(OUTS) SKIP(LINES1-LINEH1);
                           GO TO DEF;
                           END;
        PUT FILE (OUTS) EDIT ('2',SUB,UNDER)
          (LINE(LINES1-1),X((SZ-MARG+1-LS)/2+MARG-1+LS),A(1),LINE(
          LINES1),X((SZ-MARG+1-LS)/2+MARG-1),A(LS),SKIP(0),X((SZ-MARG+1
          -LS)/2+MARG-1),A(LS));
        PUT FILE (OUTS) SKIP (1);


/*  MAKE PREPARATIONS FOR SORT AND CALL THE SORT ROUTINE.  PRINT THE
    RETURN CODE.                                                      */


DEF:    SFIELD = ' SORT FIELDS=(4,7,CH,A),SIZE=E600 ';
        RFIELD = ' RECORD TYPE=F,LENGTH=(20) ';
SORT:   CALL IHESRTA(SFIELD,RFIELD,45056,RETCDE);
        PUT EDIT ('RETURN CODE IS ',RETCDE)
          (SKIP(3),A(15),F(3));


/*  PUT DATA INTO A COLUMN UNTIL THE COLUMN IS FULL.                  */



        NN = 1;
        IND = 1;
        OPEN FILE (SORTOUT);
LOOP1:  DO I = 1 TO L;
LOOP2:  DO J = 1 TO NPB;
        READ FILE (SORTOUT) INTO (STR);
        GET STRING (STR) EDIT (VT,ED) (A(10),A(10));
        IF IMP = 1 THEN PUT STRING (COL#(IND,NN)) EDIT (VT,ED)
                           (A(10),X(WIDTH-20),A(10));
                    ELSE PUT STRING (COL#(IND,NN)) EDIT (ED,VT)
                           (A(10),X(WIDTH-20),A(10));
        NN = NN + 1;
        END LOOP2;
        NN = NN + 1;
        END LOOP1;


/*  UP THE COLUMN SUBSCRIPT SO A NEW COLUMN CAN BE FILLED.  IF ALL
    THE COLUMNS, HOWEVER, HAVE BEEN FILLED FOR A PAGE, WE MUST PUT
    THE PAGE ON THE OUTPUT FILE FIRST.  PRINT A NEW TITLE ON THE
    NEXT PAGE AND INITIALIZE AS NECESSARY.  THEN FILL A NEW COLUMN.   */


         IND = IND + 1;
         IF IND = (COL+1)
           THEN DO;
                IND = 1;
```

```
               LOOP3: DO I = 1 TO L * NPB + L - 1;
                      PUT FILE (OUTS) SKIP (1);
               LOOP4: DO J = 1 TO COL;
                      PUT FILE (OUTS) EDIT (COL#(J,I))
                        (SKIP(0),COLUMN(MG(J)),A(WIDTH));
                      COL#(J,I) = ' ';
                      END LOOP4;
                      END LOOP3;
               IF INDX = 'A' THEN GO TO NO_WR;


                         /* THIS IS THE FOOTNOTE THAT APPERS
                            ON THE TITLE PAGE OF THE QUARTERLY
                            INDEX.                              */


               IF K = PAGE THEN
                 PUT FILE (OUTS) EDIT (UNDER,'1','THIS INDEX IS CUMUL',
                 'ATED ANNUALLY BEGINNING WITH THE FALL ISSUE OF ARM.',
                 '2','ANY DOCUMENT WITH AN MP NUMBER IS AVAILABLE FROM'
                 ,' EDRS BY USING THE','APPROPRIATE MP NUMBER.')
                 (SKIP(SPACE),COLUMN(MG(1)-3),A(37),SKIP(1),COLUMN(MG
                 (1)-1),A(1),SKIP(1),COLUMN(MG(1)),A(19),A(51),SKIP(1),
                 COLUMN(MG(1)-1),A(1),SKIP(1),COLUMN(MG(1)),A(48),A(18)
                 ,SKIP(1),COLUMN(MG(1)),A(22));
               NO_WR:  L = NBR;
               PAGE = PAGE + 1;
               IF SKPP = MARG  THEN DO;
                                    SKPP = SZ - 3;
                                    SKPH = MARG;
                                    END;
                               ELSE DO;
                                    SKPH = SZ - LH + 1;
                                    SKPP = MARG;
                                    END;
               PUT FILE (OUTS) EDIT (PAGE,HEAD,TITL)
                 (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(
                 SKPH),A(LH),LINE(LINET),X((SZ-MARG+1-LT)/2+MARG-1),A(
                 LT));
               IF INDX = 'A' THEN DO;
                                  PUT FILE (OUTS) SKIP(LINES2-LINET);
                                  NN = 1;
                                  GO TO LOOP1;
                                  END;
               PUT FILE (OUTS) EDIT (SUB,UNDER)
                 (LINE(LINES2),X((SZ-MARG+1-LS)/2+MARG-1),A(LS),SKIP(0)
                 ,X((SZ-MARG+1-LS)/2+MARG-1),A(LS));
               PUT FILE (OUTS) SKIP(1);
               END;
        NN = 1;
        GO TO LOOP1;


 /*  CLOSE ALL FILES AND WRITE OUT THE LAST PAGE NUMBER USED IN THE
     INDEX PRINTOUT.                                                     */


STOP:   PUT FILE (OUTS) PAGE;
        CLOSE FILE (OUTS);
        PAGE = PAGE - 1;
        PUT FILE (SYSPRINT) EDIT ('LAST PAGE USED IS ',PAGE)
          (SKIP(3),A(18),F(4));
```

```
END WORK;

END PRNTCV;
```

SUBJECT ANNUAL

```
/****************************                                          */
/*                                                                     */
/*     A N N U A L   S U B J E C T   I N D E X   P R O G R A M         */
/*                                                                     */
/*  PROGRAMMER:  RONNIE ROSEN                                          */
/*                                                                     */
/*  PURPOSE:  THE PROGRAM PRODUCES THE ANNUAL SUBJECT INDEX.  IT       */
/*            TAKES THE 8 DATA SETS FROM THE QUARTERLY OUTPUT AND      */
/*            CONCATENATES THEM TO FORM ONE LARGE DATA SET.  THE       */
/*            SORT/MERGE SORTS THESE RECORDS AND WE PRINT THEM UNDER   */
/*            SPECIFIED FORMAT TO FORM THE INDEX PRINTOUT.  THE        */
/*            SORTED DATA IS DELETED, BUT THE 8 DATA SETS WHICH ARE    */
/*            ON 2 TAPES, ARE KEPT.                                    */
/*                                                                     */
/*  DEFINITION OF VARIABLES:                                           */
/*      1. BLK -- NUMBER OF PIECES INTO WHICH THE SUBJECT KEYWORD      */
/*                MUST BE BROKEN UP SO THAT IT WILL FIT IN A COLUMN    */
/*                WIDTH.                                               */
/*      2. CNT -- THE NUMBER OF DOCUMENTS WRITTEN ABOUT SOME KEYWORD   */
/*                IN THE YEAR -P-.                                     */
/*      3. COL -- NUMBER OF COLUMNS IN A PRINT-OUT PAGE.               */
/*      4. DUMMY -- INPUT OR OUTPUT STRING USED TO CREATE OR RETRIEVE  */
/*                  RECORDS.                                           */
/*      5. FV -- THE VT NUMBERS OF THE DOCUMENTS WRITTEN ABOUT SOME    */
/*               KEYWORD  IN THE YEAR -P-.                             */
/*      6. FRST -- THE PAGE NUMBER OF THE TITLE PAGE.                  */
/*      7. FVT -- A SINGLE VT NUMBER.                                  */
/*      8. HEAD -- PART OF HEADING OF EVERY PAGE OF THE INDEX WHICH    */
/*                 LOOKS LIKE 'AIM/ARM 69-70'.                         */
/*      9. IHESRTA -- PL/1 CALL TO SORT.                               */
/*     10. IMAX -- INDICATES WHEN ALL COLUMNS OF A PAGE ARE FULL.      */
/*     11. IMP -- INDICATES ALL DATA HAS BEEN PUT INTO INDEX FORM      */
/*                AND THE SORTED FILE HAS BEEN READ ALL THROUGH.       */
/*     12. IND -- REFERENCES TO CURRENT COLUMN BEING FILLED.           */
/*     13. KNT -- THE NUMBER OF LINES OF OUTPUT IT TAKES TO PRINT A    */
/*                PUBLICATION YEAR AND THE VT NUMBERS OF DOCUMENTS     */
/*                WRITTEN THEN. THE VT NUMBERS ARE BROKEN UP INTO      */
/*                THOUSAND INTERVALS; 0 - 999 LISTED FIRST; 1000-1999  */
/*                LISTED NEXT, ETC.  EACH INTERVAL STARTS A NEW LINE.  */
/*     14. LIM -- THE NUMBER OF VT NUMBERS THAT COULD FIT ON ONE LINE  */
/*                OF A COLUMN.                                         */
/*     15. LINEH1 -- THE LINE NUMBER ON THE TITLE PAGE ON WHICH THE    */
/*                   PAGE NUMBER AND -HEAD- APPEAR.                    */
/*     16. LINEH2 -- SAME AS -LINEH1- EXCEPT FOR ALL SUCCEEDING PAGES.*/
/*     17. LINES1 -- LINE NUMBER ON THE TITLE PAGE DIRECTLY ABOVE THE */
/*                   STARTING LINE OF THE INDEX PRINTOUT.  -           */
/*     18. LINES2 -- SAME AS -LINES1- EXCEPT FOR ALL SUCCEEDING PAGES.*/
/*     19. LINET -- LINE NUMBER ON ALL PAGES EXCEPT THE TITLE PAGE     */
/*                  WHERE THE TITLE APPEARS.                           */
/*     20. LN -- COUNTER WHICH TELLS THE LAST LINE ON A PAGE THAT      */
/*               WAS FILLED WITH INFORMATION.  IT COUNTS HEADING AND   */
/*               INDEX, AND MAKES SURE WE DON'T EXCEDE THE MAXIMUM     */
/*               PAGE SIZE.                                            */
/*     21. LNE -- COUNTS THE NUMBER OF VT NUMBERS BEING PUT ON A LINE */
/*                SO THAT THE LIMIT IS NOT EXCEDED.                    */
/*     22. LTH -- LENGTH OF THE INDEX TITLE.                           */
/*     23. MARG -- LEFT-HAND MARGIN OF THE INDEX PAGE.                 */
/*     24. MG -- ARRAY OF COLUMN MARGINS EACH MEASURED FROM COLUMN 1. */
/*     25. MT -- DESIGNATES WHICH INTERVAL THE VT NUMBER FALLS IN.     */
/*               MT * 1000 TO MT * 1000 + 999, WHERE MT = 0, 1, 2,... */
/*               IS THE INTERVAL.                                      */
/*     26. NN -- COUNTS THE LINES AS A COLUMN IS FILLED.  THIS DOES    */
/*            NOT INCLUDE HEADING.                                     */00000630
/*      27. OUTS -- THE OUTPUT FILE FOR THE INDEX PRINTOUT.            */00000640
/*      28. P -- ARRAY OF PUBLICATION DATES FOR A KEYWORD  NAME WHEN   */00000650
/*              DOCUMENTS WERE PUBLISHED ABOUT THAT KEYWORD NAME.      */00000660
/*      29. PAGE -- PAGE NUMBER OF AN INDEX PAGE.                      */00000670
/*      30. PREV -- KEYWORD   WHICH HAS JUST HAD ITS DATA ADDED TO THE */00000680
/*                 INDEX PRINTOUT.                                     */00000690
/*      31. PUB -- A PUBLICATION YEAR FROM A SUBSET FILE RECORD.       */00000700
/*      32. QUE -- INDICATES WHETHER THE TITLE PAGE WILL BE ON THE     */00000710
/*                LEFT (L) OR RIGHT (R) SIDE OF THE PUBLICATION.       */00000720
/*      33. RETCDE -- THE CODE RETURNED BY SORT; '0' MEANS SUCCESS AND */00000730
/*                 '16' MEANS FAILURE.                                 */00000740
/*      34. RFIELD -- DEFINES THE RECORDS BEING SORTED FOR SORT.       */00000750
/*      35. SFIELD -- DEFINES WHAT PARTS OF THE RECORDS ARE USED AS    */00000760
/*                  THE SORT KEY FOR SORT.                             */00000770
/*      36. SIZE -- THE MAXIMUM LENGTH OF A PAGE, INDEX AND HEADING.   */00000780
/*      37. SKPH -- THE COLUMN ON -LINEH1- OR -LINEH2- WHERE -HEAD-    */00000790
/*                 SHOULD BEGIN TO BE PRINTED.                         */00000800
/*      38. SKPP -- THE COLUMN ON -LINEH1- OR -LINEH2- WHERE -PAGE-    */00000810
/*                 SHOULD BEGIN TO BE PRINTED.                         */00000820
/*      39. SORTIN -- THE EIGHT DATA SETS ON TAPE THAT ARE THE INPUT   */00000830
/*                  TO SORT.                                           */00000840
/*      40. SORTOUT -- OUTPUT FILE WITH THE SORTED RECORDS.            */00000850
/*      41. SUBJ -- KEYWORD NAME FROM A SUBSET FILE RECORD.            */00000860
/*      42. SZ -- RIGHT-HAND MARGIN OF A PAGE.                         */00000870
/*      43. TITLE -- INDEX TITLE.                                      */00000880
/*      44. V -- CHARACTER ARRAY OF VT NUMBERS FOR DOCUMENTS WRITTEN   */00000890
/*              ABOUT SOME KEYWORD IN SOME YEAR.                       */00000900
/*      45. VT -- DOCUMENT NUMBER FROM SOME SUBSET FILE RECORD.        */00000910
/*      46. WD -- NUMBER OF MAXIMUM SIGNIFICANT DIGITS IN A VT NUMBER. */00000920
/*      47. WIDTH -- COLUMN WIDTH IN A PRINTOUT.                       */00000930
/*      48. COL# -- REPRESENTS THE INDEX PART OF A PAGE.               */00000940
/*      49. PC -- PIECES INTO WHICH A KEYWORD IS BROKEN UP SO THAT IT  */00000950
/*               FITS INTO A COLUMN.                                   */00000960
/*      50. TEMP-OUT -- THE ACTUAL SET-UP OF HOW THE COMPUTER PREPARES */00000970
/*                     THE DATA FOR OUTPUT.  THE DATA IS A DATE AND    */00000980
/*                     ALL ITS VT NUMBERS.                             */00000990
/*                                                                     */00001000
/*  INPUT TO THE PROGRAM:                                              */00001010
/*    -SORTIN-                                                         */00001020
/*      THIS IS YOUR 8 DATA SETS SAVED FROM THE QUARTERLY RUNS,        */00001030
/*      CONCATENATED TOGETHER TO FORM ONE LARGE DATA SET TO BE         */00001040
/*      PROCESSED.                                                     */00001050
/*    -SYSIN-, INPUT ON CARDS                                          */00001060
/*      THIS IS A CONTINUOUS STREAM OF DATA ON SEVERAL CARDS.  THE     */00001070
/*      DATA IS VALUES FOR THE VARIABLES LISTED BELOW, IN THE SAME     */00001080
/*      ORDER AS BELOW.                                                */00001090
/*           -PAGE-          F(4)                                      */00001100
/*           -SIZE-          F(4)                                      */00001110
/*           -WIDTH-         F(4)                                      */00001120
/*           -WD-            F(4)                                      */00001130
/*           -MARG-          F(4)                                      */00001140
/*           -SZ-            F(4)                                      */00001150
/*           -LINEH1-        F(4)                                      */00001160
/*           -LINES1-        F(4)                                      */00001170
/*           -LINEH2-        F(4)                                      */00001180
/*           -LINET-         F(4)                                      */00001190
/*           -LINES2-        F(4)                                      */00001200
/*           -COL-           F(4)                                      */00001210
/*           -MG- ARRAY      COL * F(4)                                */00001220
/*           -LTH-           F(4)                                      */00001230
/*           -TITLE-         A(LTH)                                    */00001240
/*           -HEAD-           A(13)                                     */00001250
/*           -QUE-            A(1)                                      */00001260
/*      THESE VARIABLES GIVE THE OUTPUT SPECIFICATIONS AND ARE          */00001270
/*      DEFINED INDIVIDUALLY IN THE PRECEDING SECTION.  THEIR VALUES    */00001280
/*      CAN BE OBTAINED FROM THE TYPED WRITE-UP AND THE VOCATIONAL      */00001290
/*      CENTER PERSONNEL.                                               */00001300
/*                                                                      */00001310
/*  SPECIAL NOTE:                                                       */00001320
/*    PRINT THE ANNUAL INDEX ON PLAIN PAPER UNTIL IT HAS BEEN           */00001330
/*    APPROVED BY THE VOCATIONAL CENTER.  THEN IT CAN BE PRINTED ON     */00001340
/*    MAGNETIC TAPE AND THE TAPE LISTED ON THE 360/50 WITH THE TN       */00001350
/*    CHAIN FOR AN UPPER AND LOWER CASE INDEX PRINTOUT.                 */00001360
/*                                                                      */00001370
/***********************************************************************/00001380
                                                                        00001390
ANNSUB:  PROCEDURE OPTIONS (MAIN);                                      00001400
        DECLARE                                                         00001410
          BLK                FIXED BINARY (31,0),                       00001420
          CNT(15)            FIXED BINARY (31,0),                       00001430
          COL                FIXED BINARY (31,0),                       00001440
          DUMMY              CHARACTER (208),                           00001450
          FV(15,75)          FIXED BINARY (31,0) INITIAL ((1125)0),     00001460
          FRST               FIXED BINARY (31,0),                       00001470
          FVT                FIXED BINARY (31,0),                       00001480
          HEAD               CHARACTER (13),                            00001490
          I                  FIXED BINARY (31,0),                       00001500
          IHESRTA            ENTRY(CHARACTER(59),CHARACTER(28),         00001510
                                     FIXED BINARY (31,0),FIXED BINARY   00001520
                                     (15,0)),                           00001530
          IMAX               FIXED BINARY (31,0),                       00001540
          IMP                FIXED BINARY (31,0),                       00001550
          IND                FIXED BINARY (31,0),                       00001560
          J                  FIXED BINARY (31,0),                       00001570
          K                  FIXED BINARY (31,0),                       00001580
          KNT                FIXED BINARY (31,0),                       00001590
          L                  FIXED BINARY (31,0),                       00001600
          LIM                FIXED BINARY (31,0),                       00001610
          LINEH1             FIXED BINARY (31,0),                       00001620
          LINEH2             FIXED BINARY (31,0),                       00001630
          LINES1             FIXED BINARY (31,0),                       00001640
          LINES2             FIXED BINARY (31,0),                       00001650
          LINET              FIXED BINARY (31,0),                       00001660
          LN                 FIXED BINARY (31,0),                       00001670
          LNE                FIXED BINARY (31,0),                       00001680
          LTH                FIXED BINARY (31,0),                       00001690
          MARG               FIXED BINARY (31,0),                       00001700
          MG(3)              FIXED BINARY (31,0),                       00001710
          MT                 FIXED BINARY (31,0),                       00001720
          N                  FIXED BINARY (31,0),                       00001730
          NN                 FIXED BINARY (31,0),                       00001740
          OUTS               FILE STREAM,                               00001750
          P(15)              CHARACTER (2),                             00001760
          PAGE               FIXED BINARY (31,0),                       00001770
          PREV               CHARACTER (132) VARYING,                   00001780
          PUB                CHARACTER (2),                             00001790
          QUE                CHARACTER (1),                             00001800
          RETCDE             FIXED BINARY (15,0),                       00001810
          RFIELD             CHARACTER (28),                            00001820
          SFIELD             CHARACTER (59),                            00001830
          SIZE               FIXED BINARY (31,0),                       00001840
          SKPH               FIXED BINARY (31,0),                       00001850
          SKPP               FIXED BINARY (31,0),                       00001860
          SORTIN            FILE SEQUENTIAL RECORD,
          SORTOUT           FILE SEQUENTIAL RECORD,
          SUBJ              CHARACTER (200) VARYING,
          SZ                FIXED BINARY (31,0),
          TITLE             CHARACTER (50) VARYING,
          V(15,75)          CHARACTER (6) VARYING,
          VT                CHARACTER (6) VARYING,
          WD                FIXED BINARY (31,0),
          WIDTH             FIXED BINARY (31,0),
        ;



/*   SORT THE 8 DATA SETS, FIRST BY SUBJECT NAME, THEN BY PUBLICATION*/
/*   DATE, AND FINALLY BY VT NUMBER, ALL IN ASCENDING ORDER.  PRINT  */
/*   THE RETURN CODE, AN INDICATION OF SUCCESS.  WRITE OUT THE SORTED*/
/*   FILE AS A CHECK IN CASE OF TROUBLE.                              */


SFIELD=' SORT FIELDS=(1,200,CH,A,201,2,CH,A,203,6,CH,A),SIZE=E9999 ';
         RFIELD=' RECORD TYPE=F,LENGTH=(208) ';
        CALL IHESRTA (SFIELD,RFIELD,45056,RETCDE);
        PUT FILE (SYSPRINT) EDIT (RETCDE)  (SKIP(3),F(3));


/*   READ IN THE OUTPUT SPECIFICATIONS FOR THE INDEX PRINTOUT.        */


        GET FILE (SYSIN) EDIT (PAGE,SIZE,WIDTH,WD,MARG,SZ,LINEH1,LINES1
          ,LINEH2,LINET,LINES2,COL,(MG(I) DO I = 1 TO COL),LTH,TITLE,
          HEAD,QUE)
          (13 F(4),(COL) F(4),A(LTH),A(13),A(1));


/*   PRINTING IS DONE IN THIS BLOCK.                                  */


BUSINESS:
        BEGIN;
        DECLARE
          COL#(COL,SIZE-LINES2)  CHARACTER (WIDTH),
          HEADR             ENTRY (FIXED BINARY (31,0)),
          PC(4)             CHARACTER (WIDTH) VARYING,
          TEMP_OUT(SIZE-LINES2)  CHARACTER (WIDTH) VARYING,
        ;


/*   WHEN THE FILE OF SORTED RECORDS HAS BEEN FULLY READ AND PUT INTO*/
/*   THE INDEX, WE TRANSFER HERE.  THE ASSIGNMENTS MADE HERE WILL     */
/*   ALLOW TRANSFER BACK TO THE FLOW OF ACTION, WHERE THE SET-UP FOR */
/*   THE LAST PAGE CAN BE PRINTED, AND WILL CAUSE PROGRAM             */
/*   TERMINATION.                                                     */


        ON ENDFILE (SORTOUT) BEGIN;
          IMP = 1;
          SUBJ = 'ZZ';
          GO TO TEST;
          END;


/*   MAKE INITIALIZATIONS FOR PRINTING OF THE INDEX.                  */

```

```

BLKOT:  DO I = 1 TO SIZE-LINES2;
        DO J = 1 TO COL;
        COL#(J,I) = ' ';
        END;
        END BLKOT;
        OPEN FILE (SORTOUT),
             FILE (OUTS) OUTPUT STREAM PRINT PAGESIZE(SIZE + 4)
        ;
        IMP = 0;
        IND = 1;
        FRST = PAGE;
        IMAX = COL + 1;
        LIM = (WIDTH - 4)/(WD + 2);
        PUT DATA (LIM);
        NN = 0;
        LN = 9;
        IF QUE = 'L' THEN DO;
                          SKPP = MARG;
                          SKPH = SZ - 12;
                          END;
                     ELSE DO;
                          SKPP = SZ - 3;
                          SKPH = MARG;
                          END;


/*   PRINT THE HEADING FOR THE TITLE PAGE OF THE INDEX ON THE OUTPUT */
/*   FILE.                                                           */


        PUT FILE (OUTS) EDIT (PAGE,HEAD)
         (PAGE,LINE(LINEH1),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
         A(13));
        PUT FILE (OUTS) SKIP (LINES1-LINEH1);


/*   READ IN THE VERY FIRST SORTED RECORD AND BREAK THE RECORD UP    */
/*   INTO ITS INDIVIDUAL PIECES OF DATA.                             */


READ:   READ FILE (SORTOUT) INTO (DUMMY);
        GET STRING (DUMMY) EDIT (SUBJ,PUB,FVT)
             (A(200),A(2),F(6,0));
        GET STRING (DUMMY) EDIT (VT)
             (X(202),A(6));
        CALL DELET;
        N = INDEX(SUBJ,'"');
        IF N = 0 THEN PUT EDIT ('***NO END OF SUBJECT NAME IN ',SUBJ,
                       ' IN DOCUMENT ',VT,'***')
                        (SKIP(2),A(29),A(200),A(13),A(6),A(13));
        SUBJ = SUBSTR(SUBJ,1,N-1);


/*   EVERY TIME A NEW KEYWORD'S DATA IS STARTED, WE BRANCH HERE.     */
/*   INITIALIZATIONS ARE MADE IN THE ARRAYS THAT STORE ALL THE DATA  */
/*   FOR THE KEYWORD.  ARRAY -P- WILL HOLD THE PUBLICATION DATES,    */
/*   -CNT- WILL TELL HOW MANY DOCUMENTS WERE PUBLISHED UNDER EACH    */
/*   DATE, -V- AND -FV- WILL HOLD ALL DOCUMENT NUMBERS FOR EACH      */
/*   DATE, AND -I- WILL HOLD THE NUMBER OF DATES IN -P-,  -PREV- IS  */
/*   THE KEYWORD NAME.                                               */

```

```

INITL:  I = 1;
        J = 1;
        PREV = SUBJ;
        IF PUB = '  ' THEN PUB = 'ND';
        CNT(1) = 1;
        P(1) = PUB;
        V(1,1) = VT;
        FV(1,1) = FVT;


/*   READ IN A RECORD FROM THE SORTED FILE AND BREAK IT UP INTO ITS  */
/*   INDIVIDUAL DATA.                                                 */


RD_ON:  READ FILE (SORTOUT) INTO (DUMMY);
        GET STRING (DUMMY) EDIT (SUBJ,PUB,FVT)
             (A(200),A(2),F(6,0));
        GET STRING (DUMMY) EDIT (VT)
             (X(202),A(6));
        CALL DELET;
        IF PUB = '  ' THEN PUB = 'ND';
        N = INDEX(SUBJ,'"');
        IF N = 0 THEN PUT EDIT ('***NO END OF SUBJECT NAME IN ',SUBJ,
                ' IN DOCUMENT ',VT,'***')
                  (SKIP(2),A(29),A(200),A(13),A(6),A(13));
        SUBJ = SUBSTR(SUBJ,1,N-1);


/*   IF THIS NEW RECORD'S DATA BELONGS WITH THE KEYWORD WE ARE        */
/*   PROCESSING, ADD THE DATA TO THE ARRAYS.                          */


TEST:   IF SUBJ = PREV
          THEN DO;


           /* IF THE PUBLICATION DATE IS THE SAME AS THE       */
           /* PREVIOUS RECORD, ADD ANOTHER VT NUMBER TO        */
           /* THAT DATE.                                       */


                IF P(I) = PUB THEN DO;
                                   J = J + 1;
                                   CNT(I) = J;
                                   V(I,J) = VT;
                                   FV(I,J) = FVT;
                                   END;


           /* IF THE PUBLICATION DATE IS DIFFERENT, START      */
           /* A PLACE FOR A NEW DATE WITH THIS VT NUMBER       */
           /* AS THE FIRST NUMBER UNDER IT.                    */


                              ELSE DO;
                                   I = I + 1;
                                   P(I) = PUB;
                                   J = 1;
                                   V(I,1) = VT;
                                   FV(I,J) = FVT;
                                   CNT(I) = 1;
```

```
                                  END;


           /* GO BACK AND READ THE NEXT RECORD.            */


               GO TO RD_ON;
               END;


/*   IF THIS NEW RECORD'S DATA STARTS THE NEXT KEYWORD, WE MUST ADD  */
/*   THE OLD KEYWORD'S DATA TO THE INDEX PAGE BEING FILLED.  THEN    */
/*   WE WILL START THE SAME PROCEDURE OVER AGAIN FOR THE NEW KEYWORD.*/


          ELSE DO;
               IF I>15 THEN PUT EDIT ('***MORE THAN 15 DATES UNDER-- ',
                  PREV,' ***')(SKIP(2),A(30),A(200),A(4));


           /* BREAK THE KEYWORD NAME INTO SMALLER PIECES IF */
           /* IT EXCEDES THE COLUMN WIDTH IN LENGTH.        */


               L = LENGTH(PREV);
               IF L>WIDTH THEN CALL BREAK(PREV,WIDTH,PC,BLK);
                          ELSE DO;
                               BLK = 1;
                               PC(1) = PREV;
                               END;


           /* FOR EACH PUBLICATION DATE, WE WILL ADD THE    */
           /* DATE AND ALL VT NUMBERS TO THE INDEX PAGE, AND*/
           /* THEN DO THE SAME FOR THE NEXT DATE.           */


               LOOP1: DO J=1 TO I;
                      IF CNT(J)>75 THEN PUT EDIT
                        ('***MORE THAN  75 DOCUMENTS UNDER ',P(J),
                        ' FOR KEYWORD ',PC(1),' ***')
                        (SKIP(2),A(33),A(2),A(13),A(WIDTH),A(4));
                      KNT = 1;
                      TEMP_OUT(1) = P(J) || ': ';
                      LNE = 0;
                      MT = FV(J,1)/1000;


                  /* FIRST THE DATE AND VT NUMBERS,    */
                  /* BROKEN UP INTO THOUSAND INTERVALS,*/
                  /* ARE PUT INTO A TEMPORARY ARRAY    */
                  /* (TEMP_OUT).  THEIR APPEARANCE IN  */
                  /* THIS ARRAY IS IDENTICAL TO THEIR  */
                  /* APPEARANCE IN THE INDEX.  IT TOOK */
                  /* -KNT- LINES IN -TEMP_OUT- TO GET  */
                  /* THIS BLOCK OF DATA STORED.        */


                      LOOP2: DO L = 1 TO CNT(J);
                              IF MT¬=(FV(J,L)/1000) THEN DO;
                                                        MT=FV(J,L)/1000;
                                                        LNE=0;
```

```
                                     KNT=KNT+1;
                                     TEMP_OUT(KNT)=
                                       '    ';
                                     END;
           IF (L¬=1) & (LNE¬=0) THEN TEMP_OUT(KNT) =
             TEMP_OUT(KNT) || ',';
           TEMP_OUT(KNT)=TEMP_OUT(KNT)||' '||V(J,L);
           LNE = LNE + 1;
           IF (LNE = LIM)
            &(FV(J,L+1)/1000=MT)
            &(L ¬= CNT(J))
                       THEN DO;
                            TEMP_OUT(KNT)=TEMP_OUT(KNT
                              ) || ',';
                            KNT=KNT+1;
                            LNE = 0;
                            TEMP_OUT(KNT)='    ';
                            END;
            END LOOP2;


/* IF THIS IS THE FIRST DATE FOR THIS*/
/* KEYWORD, WRITE THE KEYWORD NAME IN*/
/* THE COLUMN BEING FILLED.          */


     IF J ¬= 1 THEN GO TO LP2;
     IF (LN+BLK+2) > SIZE THEN CALL HEADR(0);
     NN = NN + 1;
     LN = LN + 1;
     LP1: DO L = 1 TO BLK;
          COL#(IND,NN) = PC(L);
          NN = NN + 1;
          END LP1;
     LN = LN + BLK;


/* START TO ADD THE -TEMP_OUT- BLOCK */
/* TO THE COLUMN.  START NEW COLUMNS */
/* AS NECESSARY, BUT REPEAT THE      */
/* KEYWORD NAME AND THE DATE WHEN    */
/* THIS HAPPENS.  WHEN THE COMPLETE  */
/* BLOCK HAS BEEN PUT IN COLUMNS,    */
/* REPEAT THE COMPLETE PROCEDURE FOR */
/* THE NEXT DATE.                    */


      LP2: DO L = 1 TO KNT;
           IF (LN + 1) > SIZE
             THEN DO;
                  CALL HEADR(1);
                  DO N = 1 TO BLK;
                  COL#(IND,NN) = PC(N);
                  NN = NN + 1;
                  LN = LN + 1;
                  END;
                  IF LENGTH(PC(BLK))+12 > WIDTH
                    THEN DO;
                         COL#(IND,NN)='(CONTINUED)';
                         NN = NN + 1;
                         LN = LN + 1;
                         END;
```

```
                                     ELSE COL#(IND,NN-1)= PC(BLK) ||
                                             ' (CONTINUED)';
                                   IF L ¬= 1 THEN TEMP_OUT(L)= P(J) ||
                                         ': ' || SUBSTR(TEMP_OUT(L),5);
                                   END;
                            COL#(IND,NN) = TEMP_OUT(L);
                            NN = NN + 1;
                            LN = LN + 1;
                            END LP2;
                      END LOOP1;


           /* WHEN ALL THE KEYWORD'S INFORMATION HAS BEEN    */
           /* ADDED TO COLUMNS OF INDEX PRINTOUT, WE CAN     */
           /* GO BACK AND REPEAT ALL THIS FOR THE NEXT       */
           /* KEYWORD.  IF THERE ARE NO MORE KEYWORDS, DO    */
           /* NOT BRANCH BACK.                               */


               IF IMP = 0 THEN GO TO INITL;
               END;


/*   ADD THE LAST PAGE BEING FILLED TO THE OUTPUT FILE.  TERMINATE   */
/*   THE PROGRAM.                                                    */


LOOP4:  DO K = 1 TO SIZE-LINES2;
        PUT FILE (OUTS) SKIP(1);
LOOP5:  DO L = 1 TO COL;
        PUT FILE (OUTS) EDIT (COL#(L,K))
          (SKIP(0),COLUMN(MG(L)),A(WIDTH));
        END LOOP5;
        END LOOP4;
STOP:   PUT FILE (OUTS) PAGE;
        PUT DATA (PAGE);
        CLOSE FILE (SORTOUT),
              FILE (OUTS);


/*   THIS PROCEDURE IS CALLED WHEN A NEW COLUMN IS STARTED.  WHEN ALL*/
/*   COLUMNS ON A PAGE HAVE BEEN FILLED, WE PUT THE COLUMNS ON THE   */
/*   OUTPUT FILE.  A NEW HEADING IS PUT ON THE FILE   FOR THE NEW    */
/*   PAGE.  COLUMN INITIALIZATION PROCEDES AS FOR ANY NEW COLUMN.    */


HEADR:  PROCEDURE (NP);
        DECLARE
          (K,L)                 FIXED BINARY (31,0),
          NP                    FIXED BINARY (31,0),
        ;
        IND = IND + 1;
        IF IND = IMAX
          THEN DO;
               IND = 1;
               HD1: DO L = 1 TO SIZE-LINES2;
                    PUT FILE (OUTS) SKIP(1);
               HD2: DO K = 1 TO COL;
                    PUT FILE (OUTS) EDIT (COL#(K,L))
                      (SKIP(0),COLUMN(MG(K)),A(WIDTH));
                    COL#(K,L) = ' ';
                    END HD2;
                    END HD1;
               PAGE = PAGE + 1;
               IF SKPP = MARG THEN DO;
                                SKPP = SZ - 3;
                                SKPH = MARG;
                                END;
                           ELSE DO;
                                SKPP = MARG;
                                SKPH = SZ - 12;
                                END;
               PUT FILE (OUTS) EDIT (PAGE,HEAD,TITLE)
                 (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(
                 SKPH),A(13),LINE(LINET),X((SZ-LTH-MARG+1)/2+MARG-1),
                 A(LTH));
               PUT FILE (OUTS) SKIP(LINES2-LINET);
               END;
        IF FRST = PAGE THEN LN = LINES1;
                       ELSE LN = LINES2;
        NN = NP;
        IF NN = 0 THEN LN = LN - 1;
        RETURN;
        END HEADR;


/*   THIS PROCEDURE IS CALLED TO BREAK UP A STRING (BREAKE) INTO       */
/*   PIECES OF LENGTH -LNTH-.  THE PIECES ARE STORED IN -PC- AND       */
/*   THERE ARE -NU- OF THEM.                                           */


BREAK:  PROCEDURE(BREAKE,LNTH,BROKE,NU);
        DECLARE
          BREAKE              CHARACTER(200) VARYING,
          LNTH                FIXED BINARY (31,0),
          BROKE(4)            CHARACTER (WIDTH) VARYING,
          NU                  FIXED BINARY (31,0),
          L                   FIXED BINARY (31,0),
          SUM1                FIXED BINARY (31,0),
          TEMP                CHARACTER (132) VARYING,
        ;
        NU = 0;
INIT:   SUM1 = 0;
        TEMP = BREAKE;
SRCH:   L = INDEX(TEMP,' ');
        IF L = 0
          THEN DO;
               L = LENGTH(TEMP);
               SUM1 = SUM1 + L;
               IF SUM1 > LNTH
                 THEN DO;
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L-1);
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,SUM1-L+1);
                      RETURN;
                      END;
                 ELSE DO;
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,1,SUM1);
                      RETURN;
                      END;
               END;
SUM:    SUM1 = SUM1 + (L - 1);
```

```
        IF SUM1 <=LNTH THEN DO;
                            TEMP = SUBSTR(TEMP,L+1);
                            SUM1 = SUM1 + 1;
                            GO TO SRCH;
                            END;
        NU = NU + 1;
        BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L);
        BREAKE = SUBSTR(BREAKE,SUM1+2-L);
        SUM1 = 0;
        GO TO SUM;
        END BREAK;


/*   THIS PROCEDURE IS CALLED TO DELETE THE LEADING ZEROES FROM A VT */
/*   NUMBER.  THUS, '004506' BECOMES '4506'.                         */


DELET:
        PROCEDURE;
        DECLARE
          M                 FIXED BINARY (31,0),
          TEMP              CHARACTER (6) VARYING,
          BLANK             CHARACTER (1) INITIAL (' '),
        ;
        TEMP = VT;
LP:     DO M = 1 TO 6;
        IF SUBSTR(TEMP,M,1)='0' THEN DO;
                                   IF M=1 THEN VT=BLANK||SUBSTR(VT,2);
                                          ELSE VT=SUBSTR(VT,1,M-1) ||
                                                  BLANK            ||
                                                  SUBSTR(VT,M+1);
                                     END;
                                ELSE GO TO FIX;
        END LP;
FIX:    VT = SUBSTR(VT,6-ND+1,ND);
        RETURN;
        END DELET;

        END BUSINESS;

        END ANNSUB;
```

AUTHOR ANNUAL

```
/    *                                                                */
 *                                                                    *
/*           A N N U A L   A U T H O R   I N D E X                    */
/*                                                                    */
/*                                                                    */
/* PURPOSE: GENERATION OF AUTHOR INDEX FOR AIM/ARM ANNUAL INDEX       */
/*           USES AUTHOR TAPES GENERATED DURING THE YEAR FOR INPUT    */
/*           ONLY THE VT NUMBER, YEAR OF PUBLICATION, AND AUTHOR      */
/*           NAME ARE NEEDED FOR THE INDEX.                           */
/*           FOR NAMES OF INPUT PARAMETERS SEE COMMENT BELOW.         */
/*   SYSIN DATA: THE PARAMETERS TO CONTROL FORMATTING ARE READ IN     */
/*           USING A DATA FORM THUS TO READ IN THE VALUE OF MARGIN    */
/*           A CARD WITH -  MARGIN=    - IS REQUIRED.                 */
/*                                                                    */
/**********************************************************************/
YRAUT: PROCEDURE OPTIONS(MAIN);
BEGIN;
/*
  SORT THE QUARTERLY INDECIES INTO ONE DATA SET
*/
DCL IHESRTA
       ENTRY(CHAR(48),CHAR(34),FIXED BINARY(31,0),FIXED BINARY(15,0)),
       RETCDE FIXED BINARY(15,0),
       SFIELD CHAR(48),
       RFIELD CHAR(34);
SFIELD=' SORT FIELDS=(5,132,CH,A,137,10,CH,A),SIZE=E300 ';
RFIELD=' RECORD TYPE=V,LENGTH=(564,,,171) ';
CALL IHESRTA(SFIELD,RFIELD,61440,RETCDE);
PUT FILE(SYSPRINT) LIST('RETURN CODE=' ,RETCDE);
END;
DCL (MARGIN,VT_WIDTH,DATE,INDEX_PAGE,#_COLS,PRINT_LINE,PAGE_LINES,RAN,
     FIRST_INITIAL,NEXT_INITIAL) FIXED BINARY(31,0),
     VT_ED_SKIP FIXED BINARY(31,0),
      LAST_PRINT FIXED BINARY(31,0),
    COL_SPACE CHAR(20) VAR,
     TOP CHAR(4) INITIAL('LEFT'),
     HEADING CHAR (20) VAR;
/*

  OBTAIN PARAMETER FOR THIS INDEX
    MARGIN = WIDTH OF PRINT
    VT_WIDTH = COLUMNS ALLOCATED TO VT NUMBER
    DATE = LAST TWO DIGITS OF PRESENT YEAR
        PROGRAM ALLOWS FOR DOCUMENTS PUBLISHED BETWEEN 1950 AND PRES.
    INDEX_PAGE = FIRST PAGE NUMBER FOR THIS INDEX
    #_COLS = NUMBER OF COLUMNS DESIRED
    PRINT_LINE  = NUMBER OF COLUMNS ACROSS PAGE
    PAGE LINES = LENGTH OF PAGE IN NUMBER OF LINES
    HEADING = WHICH PUBLICATION IS THIS
    TOP = WHICH SIDE OF PAGE IS FIRST PAGE NUMBER
    FIRST_INITIAL = FIRST LINE OF PRINT FOR FIRST PAGE
    NEXT_INITIAL = FIRST LINE OF PRINT ON OTHER THAN FIRST PAGE
    COL_SPACE = BLANKS TO BE PLACE BETWEEN COLUMNS

*/
GET FILE(SYSIN) DATA(MARGIN,VT_WIDTH,DATE,INDEX_PAGE,#_COLS,PRINT_LINE,
   PAGE_LINES,HEADING,TOP,FIRST_INITIAL,NEXT_INITIAL,COL_SPACE) COPY;
      RAN=DATE-50;
 LAST_PRINT=0;
BEGIN;
DCL (INITIAL_LINE,COL_COUNT,IT,PG_L_OR_R,TOTAL_PUB(50:DATE+1),
      YEAR,VT#,J,I,K,LAST,#_VTS,PUBLICATIONS(RAN),K1,LN,LNA,
     AUTHOR_PUB(50:DATE+1,50),KEY,ITERS,FIRST,QUARTER,MAX_LINES) FIXED
     BINARY (31,0),
    (VT1,VT2) FIXED BINARY(31,0),
     AIM_ARM FIXED BINARY(31,0),
     AUTHOR_COUNT FIXED BINARY(31,0),
     SAVE_LINE CHAR(MARGIN),
     AUTHOR_SAVE(10) CHAR(MARGIN),
     OUT_PAGE(PAGE_LINES) CHAR(PRINT_LINE) VAR,
       YR_CHR CHAR(2),
    NAME CHAR(132),
     REC CHAR(949) VAR,
     PREVIOUS CHAR(132) INITIAL((132)' '),
     (LINE,OUT_COL(PAGE_LINES+10),SUBNAME) CHAR(MARGIN),
      BLANK_LINE CHAR(MARGIN);
DCL ISAVE FIXED BINARY(31,0);
     BLANK_LINE=' ';
     COL_COUNT=0;
OPEN FILE(AUTHORS) RECORD INPUT TITLE('SORTOUT');
OPEN FILE(OUT) PRINT PAGESIZE(PAGE_LINES) LINESIZE(PRINT_LINE);
          TOTAL_PUB=0;
     LNA=1;
     LN=1;
COL_COUNT=0;
INITIAL_LINE=FIRST_INITIAL;
PUT FILE(OUT) PAGE LINE(INITIAL_LINE);
/*
   SECTION TO PRINT TOP OF PAGE
*/
ON ENDPAGE(OUT) BEGIN;
         INITIAL_LINE=NEXT_INITIAL;
        IF TOP='LEFT' THEN DO;
                 TOP='RITE';
                  AIM_ARM=PRINT_LINE-LENGTH(HEADING)-1;
                 PG_L_OR_R=9;
                          END;
                          ELSE DO;
                 TOP='LEFT';
       AIM_ARM=9;
                 PG_L_OR_R=PRINT_LINE-5;
                          END;
         PUT FILE(OUT) PAGE LINE(2);
 PUT FILE(OUT) EDIT(INDEX_PAGE,HEADING,'PERSONAL AND INSTITUTIONAL AUTHO
 RS INDEX')(COLUMN(PG_L_OR_R),F(3),SKIP(0),COLUMN(AIM_ARM),A,SKIP(2),
 X(PRINT_LINE/2-21),A(40));
              PUT FILE(OUT) SKIP(3);
      COL_COUNT=0;
         OUT_PAGE='    ';
         INDEX_PAGE=INDEX_PAGE+1;
                                     END;
  /*
       WHEN END OF INPUT IS REACHED FINISH LAST PAGE AND PRINT
  */
  ON ENDFILE(AUTHORS) BEGIN;
       KEY=3;
       GO TO FINAL;
                            END;
  #_VTS=(MARGIN-5)/(VT_WIDTH+1);
                    KEY=1;
          OUT_PAGE='  ';
       ON CONVERSION BEGIN;
              PUT FILE(SYSPRINT) DATA(REC);
              GO TO INPUT;
```

```
                    END;
 INPUT: READ FILE(AUTHORS) INTO(REC);
 PUT FILE(SYSPRINT) EDIT(REC)(SKIP,A);
    YR_CHR=SUBSTR(REC,133,2);
/*
     CHECK YEAR POSITION FOR WRONG CHARACTERS
         IF OBVIOUS MISTAKE CORRECT IT
*/
     IF YR_CHR¬='  '&SUBSTR(YR_CHR,1,1)=' ' THEN SIGNAL CONVERSION;
     IF SUBSTR(YR_CHR,2,1)='L' THEN SUBSTR(REC,158,1)='1';
      IF YR_CHR='ND'|YR_CHR='  ' THEN DO;
                   YEAR=DATE+1;
                   GET STRING(REC) EDIT(NAME,VT1,VT2)(A(132),X(2),2 F(3)
                                );
                            END;
  ELSE GET STRING(REC) EDIT(NAME,YEAR,VT1,VT2)(A(132),F(2),2 F(3));
 IF YEAR<50 THEN DO;
                PUT FILE(SYSPRINT) DATA(REC);
                GO TO INPUT;
                  END;
        VT#=VT1*1000+VT2;
        IF PREVIOUS=NAME THEN GO TO FILL;
        IF LNA=1 THEN GO TO NOBLANK;
FINAL: OUT_COL(LNA)=BLANK_LINE;
         LNA=LNA+1;
NOBLANK: MAX_LINES=PAGE_LINES-INITIAL_LINE;
           AUTHOR_COUNT=0;
/*
    BREAK UP AUTHOR NAME TO FIT IN PRINT COLUMN
*/
   DO J=MARGIN,J+K,J+K,J+K,J+K,J+K,J+K;
        SUBNAME=SUBSTR(PREVIOUS,J-MARGIN+1,MARGIN);
        I=0;
              IF SUBSTR(SUBNAME,MARGIN,1)¬=' ' THEN DO;
              DO I=1 TO MARGIN-1;
               IF SUBSTR(SUBNAME,MARGIN-I,1)=' '|SUBSTR(SUBNAME,MARGIN-
                            I,1)='-' THEN GO TO BREAK;
                  END;
  PUT FILE(SYSPRINT) LIST(' AUTHOR NAME',NAME,'  VT# = ',VT#);
                                          END;
BREAK: SUBNAME=SUBSTR(SUBNAME,1,MARGIN-I);
       K=INDEX(SUBNAME,'  ');
        IF K=1 THEN GO TO DOCUM;
       IF K=0 THEN K=MARGIN-I;
     PUT STRING(LINE) EDIT(SUBNAME)(A(K));
/*
    SAVE AUTHOR NAME TO BE USED IF NESSESARY IN CONTINUATION COLUMN
*/
    AUTHOR_COUNT=AUTHOR_COUNT+1;
    AUTHOR_SAVE(AUTHOR_COUNT)=LINE;
     OUT_COL(LNA)=LINE;
    LNA=LNA+1;
   END;
 DOCUM: J=0;
              IF KEY=1 THEN DO;
                  KEY=0;
                  PREVIOUS=NAME;
                  GO TO FILL;
              END;
IF KEY¬=3 THEN KEY=0;
       IF LNA>MAX_LINES THEN DO;
                             LNA=LNA-AUTHOR_COUNT;
                             CALL PRINT;
                             DO LNA=1 TO AUTHOR_COUNT;
                                OUT_COL(LNA)=AUTHOR_SAVE(LNA);
                             END;
                        END;
        DO I=50 TO DATE;
         IF TOTAL_PUB(I)=0 THEN GO TO NONE;
        J=J+1;
         PUBLICATIONS(J)=I;
NONE:    END;
    IF TOTAL_PUB(DATE+1)=0 THEN DO;
                                 K=1;
                                 K1=PUBLICATIONS(1);
                                 END;
                            ELSE DO;
                                 K=DATE+1;
                                 K1=K;
                                 END;
OUTPUT: FIRST=1;
         LAST=#_VTS;
          THOUSANDS=FIXED(AUTHOR_PUB(K1,1)/1000);
    IF K=DATE+1 THEN GO TO JUMP;
         ITERS=#_VTS;
/*
    FILL IN PRINT LINE FOR A PARTICULAR YEAR
*/
      PUT STRING(LINE) EDIT(K1,':',AUTHOR_PUB(K1,1),(',',AUTHOR_PUB(K1,
      I) DO I=2 TO ITERS WHILE((FIXED(AUTHOR_PUB(K1,I)/1000)=THOUSANDS)
     &(I<=TOTAL_PUB(K1)))))(F(2),A(1),X(2),(ITERS)(F(VT_WIDTH),A(1)));
IF I-1=#_VTS&AUTHOR_PUB(K1,I)/1000=THOUSANDS THEN SUBSTR(LINE,ITERS*
(VT_WIDTH+1)+5,1)=',';
LOOP1: OUT_COL(LNA)=LINE;
        IF KEY=5 THEN DO;
           SAVE_LINE=LINE;
                   ISAVE=I;
                      KEY=0;
                   DO I=1 TO AUTHOR_COUNT-1;
                        OUT_COL(I)=AUTHOR_SAVE(I);
                   END;
                   I=INDEX(AUTHOR_SAVE(AUTHOR_COUNT),'           ');
                       IF I=0 THEN DO;
                        OUT_COL(AUTHOR_COUNT)=AUTHOR_SAVE(AUTHOR_COUNT);
                       I=AUTHOR_COUNT+1;
                         OUT_COL(I)='(CONTINUED)';
                       END;
                              ELSE DO;
                        OUT_COL(AUTHOR_COUNT)=SUBSTR(AUTHOR_SAVE(
                          AUTHOR_COUNT),1,I)||'(CONTINUED)';
                        I=AUTHOR_COUNT;
                      END;
                   QUARTER=I+1;
         IF SUBSTR(SAVE_LINE,1,2)='  ' THEN PUT STRING(SAVE_LINE)
                   EDIT(K1,':',SUBSTR(SAVE_LINE,4))(F(2),A(1),A);
                       OUT_COL(QUARTER)=SAVE_LINE;
                   LNA=I+1;
                   I=ISAVE;
                                    END;
      LN=LNA;
       LNA=LNA+1;
 /*
     IF THIS IS CONTINUATION OF AN AUTHOR IN A SECOND COLUMN THEN KEY =5
          AND THE AUTHOR NAME NEEDS TO BE PLACED AT TOP OF COLUMN
```

```
*/
      IF LNA>=MAX_LINES THEN DO;
                                   KEY=5;
                     QUARTER=I;
                                  CALL PRINT;
                            OUT_COL(1)=BLANK_LINE;
                    I=QUARTER;
                                  END;
 IF I>TOTAL_PUB(K1) THEN GO TO ENDK;
         LAST=I-1;
             THOUSANDS=FIXED(AUTHOR_PUB(K1,I)/1000);
       FIRST=LAST+1;
         IF FIRST>TOTAL_PUB(K1) THEN GO TO ENDK;
         LAST=LAST+#_VTS;
 JUMP: IF LAST>TOTAL_PUB(K1) THEN LAST=TOTAL_PUB(K1);
       ITERS=LAST-FIRST+1;
        IF J=I THEN PUT STRING(LINE) EDIT(K1,':')(F(2),A(1));
 /*
      FILL IN PRINT LINE FOR A PARTICULAR YEAR & DELETE YEAR IF CONTIUATI
 */
         PUT STRING(LINE) EDIT(AUTHOR_PUB(K1,FIRST),(',',AUTHOR_PUB(K1,I)
        DO I=FIRST+1 TO LAST WHILE(FIXED(AUTHOR_PUB(K1,I)/1000)=THOUSANDS
      )))(X(5),(ITERS)(F(VT_WIDTH),A(1)));
 IF I-1=LAST&AUTHOR_PUB(K1,I)/1000=THOUSANDS&LAST¬=TOTAL_PUB(K1) THEN
 SUBSTR(LINE,ITERS*(VT_WIDTH+1)+5,1)=',';
        IF K=DATE+1&FIRST=1 THEN SUBSTR(LINE,1,3)='ND:';
         GO TO LOOP1;
 ENDK: IF K=DATE+1 THEN K=1;
                    ELSE K=K+1;
       IF K<=J THEN DO;
                       K1=PUBLICATIONS(K);
                       GO TO OUTPUT;
                      END;
           PREVIOUS=NAME;
     IF KEY=3 THEN CALL PRINT;
 IF KEY=3 THEN GO TO AUT;
  LN=LNA;
      TOTAL_PUB=0;
 /*
      FILL UP ARRAYS FOR A PARTICULAR AUTHOR BEFORE FORMATTING A LINE
 */
 FILL: TOTAL_PUB(YEAR)=TOTAL_PUB(YEAR)+1;
         AUTHOR_PUB(YEAR,TOTAL_PUB(YEAR))=VT#;
         GO TO INPUT;
  AUT: PUT FILE(SYSPRINT) DATA(INDEX_PAGE);
 /*
      IF COLUMN HAS BEEN FILLED THIS PROCEDURE WILL BE CALLED THE COLUMN
            NEEDS TO BE FIT ON PAGE AND LEFT OVER SHIFTED TO TOP OF NEXT C
 */
 PRINT: PROCEDURE;
          COL_COUNT=COL_COUNT+1;
      LAST_PRINT=LN;
    IF COL_COUNT>1 THEN DO I=1 TO LAST_PRINT;
                OUT_PAGE(I)=OUT_PAGE(I)||COL_SPACE||OUT_COL(I);
                              END;
                          ELSE
                           DO I=1 TO LAST_PRINT;
                  OUT_PAGE(I)=OUT_PAGE(I)||'      '||OUT_COL(I):
                 END;
    IF LAST_PRINT¬=MAX_LINES THEN DO;
    IF COL_COUNT>1 THEN DO I=LAST_PRINT+1 TO MAX_LINES;
                 OUT_PAGE(I)=OUT_PAGE(I)||COL_SPACE||BLANK_LINE;
```

```
                           END;
                     ELSE
                      DO I=LAST_PRINT+1 TO MAX_LINES;
              OUT_PAGE(I)=OUT_PAGE(I)||'     '||BLANK_LINE;
             END;
  END;
      IF COL_COUNT=#_COLS THEN DO I=1 TO MAX_LINES;
                           PUT FILE(OUT) EDIT(OUT_PAGE(I))(SKIP,A);
                                 END;
      IF KEY=3&COL_COUNT¬=#_COLS THEN DO I=1 TO MAX_LINES;
                           PUT FILE(OUT) EDIT(OUT_PAGE(I))(SKIP,A);
                                  END;
       IF COL_COUNT=#_COLS|KEY=3 THEN SIGNAL ENDPAGE(OUT);
       IF KEY=3 THEN GO TO AUT;
       LNA=1;
END PRINT;
 END YRAUT;
```

SUPPORTING SERVICES ANNUAL

```
/*************************************************************
/*                                                                     */
/*     A N N U A L   S U P P O R T I N G   S E R V I C E S             */
/*                 I N D E X   P R O G R A M                           */
/*                                                                     */
/*                                                                     */
/*  PURPOSE:  THIS PROGRAM PRODUCES THE COMBINED AIM/ARM SUPPORTING    */
/*            SERVICES INDEX FOR THE ANNUAL PUBLICATION.  THE INPUT    */
/*            TO THE PROGRAM IS THE 4 QUARTERLY DATA SETS OF AIM       */
/*            (EVEN THOUGH THE QUARTERLY AIM PUBLICATIONS DO NOT       */
/*            HAVE THIS INDEX, WE RUN THE QUARTERLY PROGRAM WITH THE   */
/*            AIM DATA TO MAKE A DATA SET WITH THE SHORTENED RECORDS)  */
/*            AND THE 4 QUARTERLY ARM DATA SETS.  WE CONCATENATE       */
/*            ALL 8 TOGETHER AND SORT USING THE 360 SORT/MERGE TO      */
/*            A TEMPORARY DISK FILE.  THE DISK FILE IS PRINTED IN      */
/*            A FORMAT DICTATED BY THE VOCATIONAL CENTER ON AN OUTPUT  */
/*            FILE AND WE HAVE AN INDEX.  THE DISK FILE IS DELETED,    */
/*            BUT ALL 8 TAPE DATA SETS ARE KEPT.                       */
/*                                                                     */
/*  DEFINITION OF VARIABLES:                                           */
/*     1. AFIELD -- 3-DIGIT CODE READ FROM THE DISK RECORDS.           */
/*     2. BLK -- NUMBER OF PIECES INTO WHICH A CODE NAME MUST BE       */
/*              BROKEN UP SO THAT IT WILL FIT IN A COLUMN OF OUTPUT.  */
/*     3. CNT -- THIS IS AN ARRAY OF NUMBERS, EACH DENOTING THE        */
/*              NUMBER OF DOCUMENTS UNDER A PARTICULAR PUBLICATION     */
/*              DATE. IF P IS THE ARRAY OF DATES FOR A SUB-CODE,       */
/*              THEN UNDER DATE P(J) THERE ARE CNT(J) DOCUMENTS TO     */
/*              BE LISTED IN THE INDEX.  EACH CNT MUST BE LESS THAN    */
/*              OR EQUAL TO 200 FOR THIS PROGRAM'S CURRENT SETUP.      */
/*              IF IT EXCEDES 200, CHANGE THE DIMENSIONS ON            */
/*              VARIABLES FV AND V.                                    */
/*     4. P -- ARRAY OF PUBLICATION DATES OR YEAR NUMBERS FOR A        */
/*            SUB-CODE, WHEN DOCUMENTS WERE WRITTEN ABOUT THE SUB-     */
/*            CODE NAME.                                               */
/*     5. I -- THE NUMBER OF DATES RELATED TO A SUB-CODE.  THUS, FOR   */
/*            SOME SUB-CODE, THERE IS INFORMATION ABOUT IT IN          */
/*            DOCUMENTS IN THE YEARS P(1), P(2),..., P(I). -I- MUST    */
/*            BE LESS THAN OR EQUAL TO 15 FOR THE PROGRAM'S CURRENT    */
/*            SET-UP.  IF IT EXCEDES 15, CHANGE THE DIMENSIONS ON      */
/*            VARIABLES P, CNT, FV, V.                                 */
/*     6. FV -- THESE ARE THE DOCUMENT NUMBERS CONTAING INFORMATION    */
/*             ABOUT A SUB-CODE NAME.  -FV- IS A TWO DIMENSIONAL       */
/*             ARRAY; THE FIRST SUBSCRIPT -J- REFERENCES THE           */
/*             PUBLICATION YEAR -P(J)- OF THE DOCUMENT, THE SECOND     */
/*             SUBSCRIPT -L- REFERENCES A PARTICULAR DOCUMENT NO.      */
/*             UNDER THIS YEAR.                                        */
/*     7. V -- SAME AS -FV- ONLY FV IS NUMERIC AND V IS CHARACTER.     */
/*     8. COL -- NUMBER OF COLUMNS IN THE INDEX PRINTOUT.              */
/*     9. DUMMY -- STRING INTO WHICH A RECORD IS READ.                 */
/*    10. FLD -- ARRAY OF SUB-CODE NAMES.                              */
/*    11. FRST -- PAGE NUMBER OF TITLE PAGE.                           */
/*    12. FVT -- A SINGLE NUMERIC DOCUMENT NUMBER.                     */
/*    13. HEAD -- HEADING WHICH TOPS EVERY PAGE AND IS FOUND ON THE    */
/*                SAME LINE AS THE INDEX PAGE NUMBER.  ITS FORM IS     */
/*                'AIM/ARM 69-70'.                                     */
/*    14. IHESRTA -- CALL TO SORT PROGRAM.                             */
/*    15. IMAX -- INDICATES ALL COLUMNS OF A PAGE ARE FULL.            */
/*    16. IMP -- INDICATES IF WE HAVE PRINTED THE LAST BLOCK OF DATA   */
/*               AND THEREFORE, IF THE INDEX IS THROUGH.  IMP=1 SAYS   */
/*               WE ARE THROUGH; IMP=0 SAYS MORE DATA TO COME.         */
/*    17. IND -- INDICATES WHAT COLUMN ON A PAGE IS BEING FILLED.      */
/*    18. KNT -- IS THE NUMBER OF LINES OF DATA IN A BLOCK.  BY BLOCK*/
/*              I MEAN A PUBLICATION DATE AND ALL ITS DOCUMENT          */00000640
/*              NUMBERS LISTED BY THOUSANDS.  TO LIST BY THOUSANDS,     */00000650
/*              WE LIST ALL VT NUMBERS 0 < VT < 999. THEN STARTING      */00000660
/*              ON A NEW LINE WE LIST ALL VT'S WITH 1000<VT#<1999.      */00000670
/*              THIS CONTINUES UNTIL ALL DOCUMENT NUMBERS ARE           */00000680
/*              LISTED.                                                 */00000690
/*     19. LIM -- MAXIMUM NUMBER OF DOCUMENT NUMBERS THAT WILL FIT      */00000700
/*              IN A COLUMN OF SOME WIDTH.                              */00000710
/*     20. LINEH1 -- LINE ON THE TITLE PAGE ON WHICH THE PAGE NUMBER    */00000720
/*                   AND -HEAD- APPEAR.                                 */00000730
/*     21. LINEH2 -- SAME AS -LINEH1- BUT FOR ALL SUCCEEDING PAGES.     */00000740
/*     22. LINES1 -- LINE ON THE TITLE PAGE WHICH IS RIGHT ABOVE THE    */00000750
/*                   INDEX PRINTOUT.                                    */00000760
/*     23. LINES2 -- SAME AS -LINES1- BUT FOR ALL SUCCEEDING PAGES.     */00000770
/*     24. LINET -- LINE ON ALL PAGES WHERE THE TITLE IS PRINTED.       */00000780
/*     25. LN -- IS THE CURRENT LINE BEING FILLED ON A PAGE. IT IS A    */00000790
/*             CHECK TO MAKE SURE THE PAGE LENGTH IS NOT EXCEDED.       */00000800
/*     26. LNE -- COUNTS THE NUMBER OF VT NUMBERS ON A LINE SO THE      */00000810
/*              COUNT DOES NOT EXCEDE -LIM-, THE LIMIT.                 */00000820
/*     27. LTH -- LENGTH OF THE INDEX TITLE.                            */00000830
/*     28. MARG -- LEFT-HAND MARGIN OF AN INDEX PAGE.                   */00000840
/*     29. MG -- MARGINS OF THE COLUMN, EACH MEASURED FROM COLUMN 1.    */00000850
/*     30. MT -- DESIGNATES THE INTERVAL OF THE LAST VT NUMBER TO BE    */00000860
/*             PUT IN THE INDEX UNDER A PARTICULAR YEAR FOR SOME        */00000870
/*             CODE NAME.  MT * 1000 IS THE START OF THE INTERVAL,      */00000880
/*             WHERE MT = 0, 1, 2,... AND MT * 1000 + 999 ENDS THE      */00000890
/*             INTERVAL.                                                */00000900
/*     31. NN -- COUNTS THE LINE NUMBERS BEING FILLED IN A COLUMN.      */00000910
/*     32. OUTS -- OUTPUT FILE FOR INDEX PRINTOUT.                      */00000920
/*     33. PAGE -- PAGE NUMBER OF INDEX.                                */00000930
/*     34. PREV -- CURRENT CODE NAME WHOSE DATA HAS JUST BEEN PUT IN    */00000940
/*                 THE INDEX.                                           */00000950
/*     35. PUB -- PUBLICATION YEAR OF A RECORD ON THE SUBSET FILE.      */00000960
/*     36. QUE -- TITLE PAGE OF INDEX WILL BE ON THE LEFT ('L') OR      */00000970
/*              RIGHT ('R') SIDE OF THE PUBLICATION.                    */00000980
/*     37. RETCDE -- RETURN CODE RETURNED BY THE SORT INDICATING        */00000990
/*                   SUCCESS OR FAILURE.                                */00001000
/*     38. RFIELD -- DESCRIBES TO SORT THE RECORDS OF THE SUBSET FILE.*/00001010
/*     39. SFIELD -- TELLS SORT WHAT PART(S) OF THE RECORDS TO SORT     */00001020
/*                   UNDER.                                             */00001030
/*     40. SZ   -- RIGHT-HAND MARGIN OF THE PAGE.                       */00001040
/*     41. SKPH -- TELLS WHERE -HEAD- SHOULD START ON ITS OUTPUT LINE.*/00001050
/*     42. SKPP -- GIVES THE STARTING COLUMN FOR THE PAGE NUMBER ON     */00001060
/*                 ITS OUTPUT LINE.                                     */00001070
/*     43. SORTIN -- ALL 8 DATA SETS OF THE SUBSET FILE TO BE SORTED.   */00001080
/*     44. SORTOUT -- SORTED RECORDS ON DISK.                           */00001090
/*     45. SIZE -- MAXIMUM LENGTH OF OUTPUT PAGE.                       */00001100
/*     46. TITLE -- TITLE OF THE INDEX.                                 */00001110
/*     47. VT -- A DOCUMENT NUMBER FROM A RECORD OF THE SUBSET FILE.    */00001120
/*     48. WD -- NUMBER OF SIGNIFICANT DIGITS IN THE VT NUMBER.         */00001130
/*     49. WIDTH -- COLUMN WIDTH.                                       */00001140
/*     50. COL# -- IS A 2-DIMENSIONAL ARRAY REPRESENTING A PAGE OF      */00001150
/*                 OUTPUT FROM BELOW THE HEADING.  THE FIRST SUBSCRIPT*/00001160
/*                 CALLS A COLUMN AND THE SECOND SUBSCRIPT CALLS A      */00001170
/*                 LINE IN THE COLUMN.                                  */00001180
/*     51. PC -- PIECES INTO WHICH A CODE NAME IS BROKEN UP SO THAT     */00001190
/*               IT WILL FIT IN A COLUMN.                               */00001200
/*     52. TEMP_OUT -- FOR SOME SUB-CODE, THIS IS THE OUTPUT SET-UP     */00001210
/*                     FOR ALL THE VT NUMBERS UNDER ONE PUBLICATION     */00001220
/*                     DATE. -TEMP_OUT- IS PUT INTO -COL#- ONCE THE     */00001230
/*                     BLOCK IS COMPLETE.                               */00001240
/*                                                                      */00001250
/*  INPUT TO THE PROGRAM:                                               */
/*    -SORTOUT-, ALL RECORDS FROM THE SUBSET FILE IN SORTED ORDER       */
/*      WE LIST THIS FILE IN A FORMAT AND AN ANNUAL INDEX IS PRODUCED*/
/*      FOR US.                                                         */
/*    -SYSIN-, INPUT ON CARDS                                           */
/*      THIS IS A CONTINUOUS STREAM OF DATA.  THE VARIABLES BELOW ARE*/
/*      GIVEN VALUES ON THE CARDS IN THE SAME ORDER AS THEY ARE         */
/*      LISTED BELOW.                                                   */
/*             -PAGE-          F(4)                                     */
/*             -SIZE-          F(4)                                     */
/*             -WIDTH-         F(4)                                     */
/*             -WD-            F(4)                                     */
/*             -MARG-          F(4)                                     */
/*             -SZ-            F(4)                                     */
/*             -LINEH1-        F(4)                                     */
/*             -LINES1-        F(4)                                     */
/*             -LINEH2-        F(4)                                     */
/*             -LINET-         F(4)                                     */
/*             -LINES2-        F(4)                                     */
/*             -COL-           F(4)                                     */
/*             -MG- ARRAY      COL * F(4)                               */
/*             -LTH-           F(4)                                     */
/*             -TITLE-         A(LTH)                                   */
/*             -HEAD-          A(13)                                    */
/*             -QUE-           A(1)                                     */
/*      THESE VARIABLES WERE DEFINED IN THE PREVIOUS SECTION AND        */
/*      THEIR VALUES CAN BE OBTAINED FROM THE TYPED WRITE-UP OR THE     */
/*      VOCATIONAL CENTER.  THE VARIABLES DEFINE AN OUTPUT PAGE.        */
/*                                                                      */
/*  SPECIAL NOTE:                                                       */
/*      PRINT THE INDEX ON THE PRINTER UNTIL IT HAS BEEN CHECKED BY     */
/*      THE VOCATIONAL CENTER.  THEN PRINT IT ON MAGNETIC TAPE AND      */
/*      LIST THE TAPE ON THE 360/50 USING THE TN CHAIN FOR UPPER AND    */
/*      LOWER CASE LETTERS.  THIS IS THE GOOD COPY OF THE INDEX.        */
/*                                                                      */
/************************************************************************/

ANNVCT:  PROCEDURE OPTIONS (MAIN);
     DECLARE
       AFIELD              FIXED BINARY (31,0),
       BLK                 FIXED BINARY (31,0),
       CNT(15)             FIXED BINARY (31,0),
       COL                 FIXED BINARY (31,0),
       DUMMY               CHARACTER (18),
       FLD(0:13)           CHARACTER (132) VARYING INITIAL
                           ('AGRICULTURAL EDUCATION',
                            'BUSINESS AND OFFICE EDUCATION',
                            'DISTRIBUTIVE EDUCATION',
                            'HEALTH OCCUPATIONS EDUCATION',
                            'HOME ECONOMICS EDUCATION',
                            'INDUSTRIAL ARTS EDUCATION',
                            'MANPOWER ECONOMICS',
                            'OCCUPATIONAL PSYCHOLOGY',
                            'OCCUPATIONAL SOCIOLOGY',
                            ' ',
                            'TRADE AND INDUSTRIAL EDUCATION',
                            ' ',
                         'GENERAL VOCATIONAL AND TECHNICAL EDUCATION',
                            'TECHNICAL EDUCATION'),
       FRST                FIXED BINARY (31,0),
       FV(15,200)          FIXED BINARY (31,0) INITIAL ((3000)0),
       FVT                 FIXED BINARY (31,0),
      HEAD                CHARACTER (13),                                 00001880
      I                   FIXED BINARY (31,0),                            00001890
      IHESRTA             ENTRY(CHARACTER(53),CHARACTER(27),              00001900
                               FIXED BINARY(31,0),FIXED BINARY            00001910
                               (15,0)),                                   00001920
      IMAX                FIXED BINARY (31,0),                            00001930
      IMP                 FIXED BINARY (31,0),                            00001940
      IND                 FIXED BINARY (31,0),                            00001950
      J                   FIXED BINARY (31,0),                            00001960
      K                   FIXED BINARY (31,0),                            00001970
      KNT                 FIXED BINARY (31,0),                            00001980
      L                   FIXED BINARY (31,0),                            00001990
      LIM                 FIXED BINARY (31,0),                            00002000
      LINEH1              FIXED BINARY (31,0),                            00002010
      LINEH2              FIXED BINARY (31,0),                            00002020
      LINES1              FIXED BINARY (31,0),                            00002030
      LINES2              FIXED BINARY (31,0),                            00002040
      LINET               FIXED BINARY (31,0),                            00002050
      LN                  FIXED BINARY (31,0),                            00002060
      LNE                 FIXED BINARY (31,0),                            00002070
      LTH                 FIXED BINARY (31,0),                            00002080
      MARG                FIXED BINARY (31,0),                            00002090
      MG(3)               FIXED BINARY (31,0),                            00002100
      MT                  FIXED BINARY (31,0),                            00002110
      N                   FIXED BINARY (31,0),                            00002120
      NN                  FIXED BINARY (31,0),                            00002130
      OUTS                FILE STREAM,                                    00002140
      P(15)               CHARACTER (2),                                  00002150
      PAGE                FIXED BINARY (31,0),                            00002160
      PREV                CHARACTER (132) VARYING,                        00002170
      PUB                 CHARACTER (2),                                  00002180
      QUE                 CHARACTER (1),                                  00002190
      RETCDE              FIXED BINARY (15,0),                            00002200
      RFIELD              CHARACTER (27),                                 00002210
      SFIELD              CHARACTER (53),                                 00002220
      SIZE                FIXED BINARY (31,0),                            00002230
      SKPH                FIXED BINARY (31,0),                            00002240
      SKPP                FIXED BINARY (31,0),                            00002250
      SORTIN              FILE SEQUENTIAL RECORD,                         00002260
      SORTOUT             FILE SEQUENTIAL RECORD,                         00002270
      SZ                  FIXED BINARY (31,0),                            00002280
      TITLE               CHARACTER (50) VARYING,                         00002290
      V(15,200)           CHARACTER (6) VARYING,                          00002300
      VT                  CHARACTER (6) VARYING,                          00002310
      WD                  FIXED BINARY (31,0),                            00002320
      WIDTH               FIXED BINARY (31,0),                            00002330
    ;                                                                     00002340
                                                                          00002350
                                                                          00002360
/*  SORT THE 8 DATA SETS OF THE SUBSET FILE BY ASCENDIND ORDER SUB-COD00002370
    THEN BY ASCENDING ORDER PUBLICATION YEAR, AND FINALLY BY ASCENDING00002380
    ORDER VT NUMBER.  PRINT OUT THE RETURN CODE AND THE SORTED RECORDS00002390
    FOR A CHECK.                                                    */00002400
                                                                          00002410
                                                                          00002420
    SFIELD = ' SORT FIELDS=(1,3,CH,A,4,2,CH,A,6,6,CH,A),SIZE=E9999 00002430
';                                                                        00002440
    RFIELD = ' RECORD TYPE=F,LENGTH=(18) ';                               00002450
    CALL IHESRTA (SFIELD,RFIELD,45056,RETCDE);                            00002460
    PUT FILE (SYSPRINT) EDIT (RETCDE)  (SKIP(3),F(3));                    00002470
                                                                          00002480
                                                                          00002490
```

```
/*   READ IN THE PAGE SPECIFICATIONS OF AN INDEX PAGE.                  */


        GET FILE (SYSIN) EDIT (PAGE,SIZE,WIDTH,WD,MARG,SZ,LINEH1,LINES1
         ,LINEH2,LINET,LINES2,COL,(MG(I) DO I = 1 TO COL),LTH,TITLE,
          HEAD,QUE)
          (13 F(4),(COL) F(4),A(LTH),A(13),A(1));


/*   BEGIN PRINTING OF THE INDEX.                                       */


BUSINESS:
        BEGIN;
        DECLARE
          COL#(COL,SIZE-LINES2)  CHARACTER (WIDTH),
          HEADR                  ENTRY (FIXED BINARY (31,0)),
          PC(4)                  CHARACTER (WIDTH) VARYING,
          TEMP_OUT(SIZE-LINES2)  CHARACTER (WIDTH) VARYING,
        ;


/*   WHEN ALL OF THE SORTED RECORDS HAVE BEEN READ, WE MUST PRINT OUT
     THAT LAST PAGE BEING PUT TOGETHER.                                 */


        ON ENDFILE (SORTOUT) BEGIN;
          IMP = 1;
          FLD(K) = 'ZZ';
          GO TO TEST;
          END;


/*   MAKE INITIALIZATIONS FOR PRINTING THE INDEX.                       */


BLKOT:  DO I = 1 TO SIZE-LINES2;
        DO J = 1 TO COL;
        COL#(J,I) = ' ';
        END;
        END BLKOT;
        OPEN FILE (SORTOUT),
             FILE (OUTS) OUTPUT STREAM PRINT PAGESIZE(SIZE + 4)
        ;
        IND = 1;
        FRST = PAGE;
        IMAX = COL + 1;
        LIM = (WIDTH - 4)/(WD + 2);
        PUT DATA (LIM);
        NN = 0;
        LN = 9;
        IMP = 0;
        IF QUE = 'L' THEN DO;
                          SKPP = MARG;
                          SKPH = SZ - 12;
                          END;
                     ELSE DO;
                          SKPP = SZ - 3;
                          SKPH = MARG;
                          END;

```

```
/*   PUT A TITLE ON THE OUTPUT FILE FOR THE INDEX TITLE PAGE.           */


          PUT FILE (OUTS) EDIT (PAGE,HEAD)
            (PAGE,LINE(LINEH1),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
            A(13));
          PUT FILE (OUTS) SKIP (LINES1-LINEH1);


 /*   READ IN THE FIRST RECORD OF THE SORTED FILE.                       */


 READ:    READ FILE (SORTOUT) INTO (DUMMY);
          GET STRING (DUMMY) EDIT (AFIELD,PUB,VT)
               (F(3,0),A(2),A(6));
          GET STRING (DUMMY) EDIT (FVT)
               (X(5),F(6,0));
          CALL DELET;
          IF AFIELD = 495 THEN K = 13;
                          ELSE K = (AFIELD-400)/10;


 /*   WE WILL BE READING -SORTOUT- UNTIL ALL THE INFORMATION FOR A
      SUB-CODE IS OBTAINED.  WE MAKE CERTAIN INITIALIZATIONS FOR
      THIS.                                                              */


 INITL:   I = 1;
          J = 1;
          PREV = FLD(K);
          IF PUB = '  ' THEN PUB = 'ND';
          CNT(1) = 1;
          P(1) = PUB;
          V(1,1) = VT;
          FV(1,1) = FVT;


 /*   READ IN A RECORD FROM -SORTOUT-.                                   */


 RD_ON:   READ FILE (SORTOUT) INTO (DUMMY);
          GET STRING (DUMMY) EDIT (AFIELD,PUB,VT)
               (F(3,0),A(2),A(6));
          GET STRING (DUMMY) EDIT (FVT)
               (X(5),F(6,0));
          CALL DELET;
          IF PUB = '  ' THEN PUB = 'ND';
          IF AFIELD = 495 THEN K = 13;
                          ELSE K = (AFIELD-400)/10;


 /*   IF THE SAME SUB-CODE IS STILL BEING USED, THEN TAKE THIS PATH.    */


 TEST:    IF FLD(K) = PREV
             THEN DO;


               /* IF THE DATE IS STILL THE SAME, ADD THIS DOCUMENT
                  TO THE NEXT POSITION UNDER THAT DATE.                */
            IF P(I) = PUB THEN DO;
                            J = J + 1;
                            CNT(I) = J;
                            V(I,J) = VT;
                            FV(I,J) = FVT;
                            END;


         /* IF THE DATE IS NEW, THIS DOCUMENT TAKES THE FIRST
            POSITION UNDER THE NEW DATE, AND THE NEW DATE
            TAKES ITS PLACE IN THE LINE OF DATES.          */


                        ELSE DO;
                            I = I + 1;
                            P(I) = PUB;
                            J = 1;
                            V(I,1) = VT;
                            FV(I,J) = FVT;
                            CNT(I) = 1;
                            END;


/*   CONTINUE READING -SORTOUT-.                                */


            GO TO RD_ON;
            END;


/*   WITH ALL THE DATA FOR A SUB-CODE, WE CAN NOW PUT THIS BLOCK IN
     THE PRINTOUT.                                              */


        ELSE DO;


/*   CHECK THAT THE NUMBER OF DATES < 15 FOR THIS SUB-CODE.  BREAK
     UP THE SUB-CODE NAME IF IT IS TOO WIDE FOR THE COLUMN.     */


            IF I>15 THEN PUT EDIT ('***MORE THAN 15 DATES UNDER-- ',
              PREV,' ***') (SKIP(2),A(30),A(132),A(4));
            L = LENGTH(PREV);
            IF L>WIDTH THEN CALL BREAK(PREV,WIDTH,PC,BLK);
                     ELSE DO;
                          BLK = 1;
                          PC(1) = PREV;
                          END;


/*   WE ADD THE DATA TO THE INDEX, GOING BY ONE DATE AT A TIME.  WE
     FILL -TEMP_OUT- WITH THE DATE AND ALL THE VT NUMBERS UNDER IT.
     FIRST, PUT THE DATE P(J) INTO -TEMP_OUT-.                  */


            LOOP1: DO J=1 TO I;
                  IF CNT(J)>200 THEN PUT EDIT
                    ('***MORE THAN 200 DOCUMENTS UNDER ',P(J),
                    ' FOR KEYWORD ',PC(1),' ***')
                    (SKIP(2),A(33),A(2),A(13),A(WIDTH),A(4));
                  KNT = 1;
```

```
                       TEMP_OUT(1) = P(J) || ': ';
                       LNE = 0;
                       MT = FV(J,1)/1000;


                /* ADD THE VT NUMBERS UNDER DATE -P(J)- INTO
                   -TEMP_OUT-, BUT BREAK THEM UP BY THOUSAND
                   INTERVALS AS YOU DO THIS.  -KNT- IS THE
                   NUMBER OF ELEMENTS IN -TEMP_OUT- THIS TOOK. */


                       LOOP2: DO L = 1 TO CNT(J);
                              IF MT¬=(FV(J,L)/1000) THEN DO;
                                                         MT=FV(J,L)/1000;
                                                         LNE=0;
                                                         KNT=KNT+1;
                                                         TEMP_OUT(KNT)=
                                                           '    ';
                                                         END;
                              IF (L¬=1) & (LNE¬=0) THEN TEMP_OUT(KNT) =
                                TEMP_OUT(KNT) || ',';
                              TEMP_OUT(KNT)=TEMP_OUT(KNT)||' '||V(J,L);
                              LNE = LNE + 1;
                              IF (LNE = LIM)
                               &(FV(J,L+1)/1000=MT)
                               &(L ¬= CNT(J))
                                        THEN DO;
                                            TEMP_OUT(KNT)=TEMP_OUT(KNT
                                              ) || ',';
                                            KNT=KNT+1;
                                            LNE = 0;
                                            TEMP_OUT(KNT)='    ';
                                            END;
                              END LOOP2;


/*    START ADDING THIS BLOCK OF DATA TO THE COLUMN ARRANGEMENT.  IF
      THIS IS THE FIRST DATE FOR THE SUB-CODE, PUT THE SUB-CODE
      NAME INTO THE COLUMN.                                           */


                       IF J ¬= 1 THEN GO TO LP2;
                       IF (LN+BLK+2) > SIZE THEN CALL HEADR(0);
                       NN = NN + 1;
                       LN = LN + 1;
                       LP1: DO L = 1 TO BLK;
                            COL#(IND,NN) = PC(L);
                            NN = NN + 1;
                            END LP1;
                       LN = LN + BLK;


                /* START ADDING THE VT NUMBERS TO THE COLUMN.
                   IF WE FINISH FILLING THE COLUMN, A NEW ONE
                   MUST BE STARTED.  THIS CAUSES THE SUB-CODE
                   NAME AND SOMETIMES THE DATE TO BE REPEATED.
                   CONTINUE UNTIL THE WHOLE BLOCK IS IN
                   COLUMNS.                                   */


                       LP2: DO L = 1 TO KNT;
                            IF (LN + 1) > SIZE
```

```
                             THEN DO;
                                  CALL HEADR(1);
                                  DO N = 1 TO BLK;
                                  COL#(IND,NN) = PC(N);
                                  NN = NN + 1;
                                  LN = LN + 1;
                                  END;
                                  IF LENGTH(PC(BLK))+12 > WIDTH
                                    THEN DO;
                                          COL#(IND,NN)='(CONTINUED)';
                                          NN = NN + 1;
                                          LN = LN + 1;
                                          END;
                                    ELSE COL#(IND,NN-1)= PC(BLK) ||
                                             ' (CONTINUED)';
                                  IF L ¬= 1 THEN TEMP_OUT(L)= P(J) ||
                                         ': ' || SUBSTR(TEMP_OUT(L),5);
                                  END;
                            COL#(IND,NN) = TEMP_OUT(L);
                            NN = NN + 1;
                            LN = LN + 1;
                            END LP2;
                      END LOOP1;


/*   IF THERE IS MORE DATA, START THIS PROCEDURE FOR THE NEXT SUB-CODE.
     OTHERWISE, PUT THE PAGE BEING FILLED ON TO THE OUTPUT FILE.      */


                IF IMP = 0 THEN GO TO INITL;
                END;
LOOP4:  DO K = 1 TO SIZE-LINES2;
        PUT FILE (OUTS) SKIP(1);
LOOP5:  DO L = 1 TO COL;
        PUT FILE (OUTS) EDIT (COL#(L,K))
          (SKIP(0),COLUMN(MG(L)),A(WIDTH));
        END LOOP5;
        END LOOP4;
STOP:   PUT FILE (OUTS) PAGE;
        PUT DATA (PAGE);
        CLOSE FILE (SORTOUT),
              FILE (OUTS);


/*   THIS PROCEDURE STARTS A NEW COLUMN.                                 */


HEADR:  PROCEDURE (NP);
        DECLARE
          (K,L)                FIXED BINARY (31,0),
           NP                  FIXED BINARY (31,0),
        ;


/*   UP THE COLUMN SUBSCRIPT FOR THE NEXT COLUMN.                        */


        IND = IND + 1;


/*   IF ALL THE COLUMNS ON THE PAGE ARE FULL, WE MUST PUT THE COLUMNS
     ON THE OUTPUT FILE.  THEN A TITLE IS ADDED TO THE OUTPUT FILE
     FOR A NEW PAGE.  WE WILL BE FILLING COLUMN 1 OF THE NEW PAGE.    */


        IF IND = IMAX
          THEN DO;
               IND = 1;
               HD1: DO L = 1 TO SIZE-LINES2;
                    PUT FILE (OUTS) SKIP(1);
               HD2: DO K = 1 TO COL;
                    PUT FILE (OUTS) EDIT (COL#(K,L))
                      (SKIP(0),COLUMN(MG(K)),A(WIDTH));
                    COL#(K,L) = ' ';
                    END HD2;
                    END HD1;
               PAGE = PAGE + 1;
               IF SKPP = MARG THEN DO;
                                SKPP = SZ - 3;
                                SKPH = MARG;
                                END;
                           ELSE DO;
                                SKPP = MARG;
                                SKPH = SZ - 12;
                                END;
               PUT FILE (OUTS) EDIT (PAGE,HEAD,TITLE)
                 (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(
                 SKPH),A(13),LINE(LINET),X((SZ-LTH-MARG+1)/2+MARG-1),
                 A(LTH));
               PUT FILE (OUTS) SKIP(LINES2-LINET);
               END;


/*   INITIALIZE LINE COUNTERS FOR THE NEW COLUMN.                     */


        IF FRST = PAGE THEN LN = LINES1;
                       ELSE LN = LINES2;
        NN = NP;
        IF NN = 0 THEN LN = LN - 1;
        RETURN;
        END HEADR;


/*   THIS PROCEDURE TAKES A STRING -BREAKE- AND BREAKS IT UP INTO
     PIECES OF LENGTH -LNTH-, STORING THEM IN -BROKE-.  -NU- IS THE
     NUMBER OF PIECES IT TOOK.                                        */


BREAK:  PROCEDURE(BREAKE,LNTH,BROKE,NU);
        DECLARE
          BREAKE            CHARACTER (132) VARYING,
          LNTH              FIXED BINARY (31,0),
          BROKE(4)          CHARACTER (WIDTH) VARYING,
          NU                FIXED BINARY (31,0),
          L                 FIXED BINARY (31,0),
          SUM1              FIXED BINARY (31,0),
          TEMP              CHARACTER (132) VARYING,
        ;
        NU = 0;
INIT:   SUM1 = 0;
        TEMP = BREAKE;
SRCH:   L = INDEX(TEMP,' ');
        IF L = 0
```

```
          THEN DO;
               L = LENGTH(TEMP);
               SUM1 = SUM1 + L;
               IF SUM1 > LNTH
                 THEN DO;
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L-1);
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,SUM1-L+1);
                      RETURN;
                      END;
                 ELSE DO;
                      NU = NU + 1;
                      BROKE(NU) = SUBSTR(BREAKE,1,SUM1);
                      RETURN;
                      END;
               END;
SUM:    SUM1 = SUM1 + (L - 1);
        IF SUM1 <=LNTH THEN DO;
                            TEMP = SUBSTR(TEMP,L+1);
                            SUM1 = SUM1 + 1;
                            GO TO SRCH;
                            END;
        NU = NU + 1;
        BROKE(NU) = SUBSTR(BREAKE,1,SUM1-L);
        BREAKE = SUBSTR(BREAKE,SUM1+2-L);
        SUM1 = 0;
        GO TO SUM;
        END BREAK;


/*   THIS PROCEDURE TAKES A VT NUMBER LIKE '003406' AND ELIMINATES
     LEADING ZEROES SO IT BECOMES '3406'.                            */


DELET:
        PROCEDURE;
        DECLARE
          M                 FIXED BINARY (31,0),
          TEMP              CHARACTER (6) VARYING,
          BLANK             CHARACTER (1) INITIAL (' '),
        ;
        TEMP = VT;
LP:     DO M = 1 TO 6;
        IF SUBSTR(TEMP,M,1)='0' THEN DO;
                                   IF M=1 THEN VT=BLANK||SUBSTR(VT,2);
                                          ELSE VT=SUBSTR(VT,1,M-1) ||
                                                  BLANK            ||
                                                  SUBSTR(VT,M+1);
                                     END;
                                ELSE GO TO FIX;
        END LP;
FIX:    VT = SUBSTR(VT,6-WD+1,WD);
        RETURN;
        END DELET;

        END BUSINESS;

        END ANNVCT;
```

DOCUMENT NUMBER ANNUAL

```
/*********************************************************************/
/*                                                                   */
/*                                                                   */
/*    A N N U A L   D O C U M E N T   N U M B E R   I N D E X        */
/*                         P R O G R A M                             */
/*                                                                   */
/*                                                                   */
/* PURPOSE:  THIS PROGRAM PRODUCES THE ANNUAL DOCUMENT NUMBER INDEX.*/
/*           IT TAKES THE FOUR ARM TAPE DATA SETS AND THE FOUR AIM   */
/*           DATA SETS AND CONCATENATES ALL 8 TOGETHER.  THEN WE     */
/*           SORT THE RECORDS AND PRINT THEM IN A FORMAT SPECIFIED   */
/*           BY THE VOCATIONAL CENTER.  THIS GIVES US AN ANNUAL      */
/*           INDEX.                                                  */
/*                                                                   */
/* DEFINITION OF VARIABLES:                                          */
/*     1. COL -- NUMBER OF COLUMNS IN THE INDEX PRINTOUT.            */
/*     2. HEAD -- HEADING AT THE TOP OF EVERY INDEX PAGE WHICH HAS   */
/*                THIS FORM:  'AIM/ARM 69-70'.                       */
/*     3. LINEH1 -- THE LINE ON THE TITLE PAGE ON WHICH -HEAD- AND   */
/*                  THE PAGE NUMBER APPEAR.                          */
/*     4. LINEH2 -- SAME AS  -LINEH1- BUT FOR ALL SUCCEEDING PAGES.  */
/*     5. LINES1 -- THE LINE ON THE TITLE PAGE ON WHICH THE SUB-TITLE*/
/*                  APPEARS.                                         */
/*     6. LINES2 -- SAME AS -LINES1- BUT FOR ALL SUCCEEDING PAGES.   */
/*     7. LINET -- LINE ON ALL PAGE BUT THE TITLE PAGE ON WHICH      */
/*                 THE INDEX TITLE APPEARS.                          */
/*     8. LT -- LENGTH OF INDEX TITLE.                               */
/*     9. MARG -- LEFT-HAND MARGIN OF THE PAGE.                      */
/*    10. MG -- ARRAY OF COLUMN MARGINS, EACH MEASURED FROM COLUMN   */
/*              1.                                                   */
/*    11. NBH -- NUMBER OF BLOCKS OF OUTPUT ON THE TITLE PAGE.  A    */
/*               BLOCK OF OUTPUT IS A FEW LINES OF OUTPUT, AND       */
/*               BLOCKS ARE SEPARATED BY A BLANK LINE.               */
/*    12. NBR -- NUMBER OF BLOCKS OF OUTPUT ON ALL OTHER PAGES.      */
/*    13. NPB -- NUMBER OF LINES OF OUTPUT IN ONE BLOCK.             */
/*    14. PAGE -- PAGE NUMBER OF AN INDEX PAGE.                      */
/*    15. QUE -- INDICATES WHETHER THE TITLE PAGE WILL BE ON THE LEFT*/
/*               (L) OR RIGHT (R) SIDE OF THE PUBLICATION.           */
/*    16. SIZE -- MAXIMUM PAGE LENGTH ALLOWED.  THIS INCLUDES HEADING*/
/*                AND INDEX ON THE PAGE.                             */
/*    17. SORTIN -- EIGHT SUBSET DATA SETS FROM QUARTERLY RUNS, ALL  */
/*                  CONCATENATED TOGETHER TO FORM ONE LARGE SET.     */
/*    18. SORTOUT -- SORTED DISK FILE.                               */
/*    19. SPACE1 -- NUMBER OF COLUMNS BETWEEN THE VT NUMBER AND      */
/*                  ISSUE IN THE PRINTOUT.                           */
/*    20. SPACE2 -- NUMBER OF COLUMNS BETWEEN THE ISSUE AND PAGE     */
/*                  NUMBER IN THE PRINTOUT.                          */
/*    21. STR -- STRING USED TO INPUT AND OUTPUT FILE RECORDS.       */
/*    22. SZ -- RIGHT-HAND MARGIN OF THE PAGE.                       */
/*    23. TITL -- TITLE OF THE INDEX.                                */
/*    24. WD -- NUMBER OF SIGNIFICANT DIGITS IN THE VT NUMBER.       */
/*    25. COL# -- REPRESENTS THE COLUMNS OF A PAGE STARTING BELOW    */
/*                THE HEADING.  THE FIRST SUBSCRIPT OF THIS MATRIX   */
/*                REFERENCES A COLUMN AND THE SECOND SUBSCRIPT       */
/*                REFERENCES A LINE IN THAT COLUMN.                  */
/*    26. IND -- THE CURRENT COLUMN BEING FILLED ON AN INDEX PAGE.   */
/*    27. ISSUE -- IS A 3 CHARACTER CODE; THE FIRST CHARACTER IS 'A' */
/*                 FOR 'AIM' OR ' ' FOR 'ARM'; THE SECOND AND THIRD  */
/*                 CHARACTERS GIVE THE SEASON 'F ','W ','SP','SU'.   */
/*    28. NN -- THE NEXT LINE TO BE FILLED IN A COLUMN OF AN INDEX   */
/*              PAGE.                                                */
/*    29. PG -- A PAGE NUMBER READ OFF A SUBSET FILE RECORD.         */
/*    30. SKPH -- COLUMN IN WHICH WE START PRINTING -HEAD-.                */
/*    31. SKPP -- COLUMN IN WHICH WE START PRINTING THE PAGE NUMBER        */
/*                OF AN INDEX PAGE.                                        */
/*    32. VT -- A VT NUMBER READ OFF OF A SUBSET FILE RECORD.              */
/*    33. YR -- THE YEAR OF THE 'AIM' OR 'ARM' ISSUE WHEN A DOCUMENT       */
/*              APPEARED IN IT.                                            */
/*    34. IHESRTA -- PL/1 CALL TO SORT.                                    */
/*    35. RETCDE -- CODE RETURNED BY THE SORT INDICATING SUCCESS OR        */
/*                  FAILURE.                                               */
/*    36. SFIELD -- INDICATES TO SORT WHAT PART OF THE RECORDS TO          */
/*                  SORT UNDER.                                            */
/*    37. RFIELD -- DEFINES FOR SORT THE TYPE OF RECORDS TO BE             */
/*                  SORTED.                                                */
/*                                                                         */
/*  INPUT TO THE PROGRAM:                                                  */
/*    -SORTIN-                                                             */
/*      THIS IS THE 8 DATA SETS, ON ONE OR TWO MAGNETIC TAPES, WHICH       */
/*      ARE SAVED AFTER EVERY QUARTERLY RUN OF AIM AND ARM.                */
/*    -SYSIN-,INPUT ON CARDS                                               */
/*      THE INPUT ON CARDS RELAYS THE OUTPUT SPECIFICATIONS TO THE         */
/*      PROGRAM.  THE DATA IS ONE CONTINUOUS STREAM ON THE CARDS           */
/*      AND IT WILL TAKE SEVERAL CARDS.  THE VALUES OF THE VARIABLES       */
/*      IN THEIR ORDER BELOW ARE THE DATA ON THE CARDS.                    */
/*            -PAGE-           F(4)                                        */
/*            -SIZE-           F(4)                                        */
/*            -WD-             F(4)                                        */
/*            -MARG-           F(4)                                        */
/*            -SZ-             F(4)                                        */
/*            -LINEH1-         F(4)                                        */
/*            -LINES1-         F(4)                                        */
/*            -LINEH2-         F(4)                                        */
/*            -LINET-          F(4)                                        */
/*            -LINES2-         F(4)                                        */
/*            -SPACE1-         F(4)                                        */
/*            -SPACE2-         F(4)                                        */
/*            -COL-            F(4)                                        */
/*            -MG- ARRAY       COL * F(4)                                  */
/*            -NPB-            F(4)                                        */
/*            -NBR-            F(4)                                        */
/*            -LT-             F(4)                                        */
/*            -TITL-           A(LT)                                       */
/*            -HEAD-           A(13)                                       */
/*            -QUE--           A(1)                                        */
/*      THESE VARIABLES WERE DEFINED IN THE PREVIOUS SECTION AND           */
/*      THEIR VALUES CAN BE OBTAINED FROM THE TYPED WRITE-UP OR THE        */
/*      VOCATIONAL CENTER.                                                 */
/*                                                                         */
/*  SPECIAL NOTE:                                                          */
/*    PRINT THE INDEX ON THE PRINTER UNTIL IT HAS BEEN APPROVED BY         */
/*    THE VOCATIONAL CENTER.  THEN IT CAN BE PRINTED ON TAPE AND THE       */
/*    TAPE CAN BE LISTED ON THE 360/50 WITH THE TN CHAIN.  THIS WILL       */
/*    GIVES US AN UPPER AND LOWER CASE LISTING.                            */
/*                                                                         */
/***************************************************************************/

ANNDOC: PROCEDURE OPTIONS (MAIN);
          DECLARE
            COL                   FIXED BINARY (31,0),
            HEAD                  CHARACTER (13),
            I                     FIXED BINARY (31,0),
            LINEH1                FIXED BINARY (31,0),
            LINEH2                FIXED BINARY (31,0),
          LINES1            FIXED BINARY (31,0),
          LINES2            FIXED BINARY (31,0),
          LINET             FIXED BINARY (31,0),
          LT                FIXED BINARY (31,0),
          MARG              FIXED BINARY (31,0),
          MG(4)             FIXED BINARY (31,0),
          NBH               FIXED BINARY (31,0),
          NBR               FIXED BINARY (31,0),
          NPB               FIXED BINARY (31,0),
          PAGE              FIXED BINARY (31,0),
          QUE               CHARACTER (1),
          SIZE              FIXED BINARY (31,0),
          SORTIN            FILE SEQUENTIAL RECORD,
          SORTOUT           FILE SEQUENTIAL RECORD,
          SPACE1            FIXED BINARY (31,0),
          SPACE2            FIXED BINARY (31,0),
          STR               CHARACTER (18),
          SZ                FIXED BINARY (31,0),
          TITL              CHARACTER (50) VARYING,
          WD                FIXED BINARY (31,0),
        ;


/* SORT THE 8 SUBSET FILES BY VT NUMBER IN ASCENDING ORDER.  PRINT    */
/* OUT THE RETURN CODE.                                                */


SORT:   BEGIN;
        DECLARE
          IHESRTA           ENTRY(,,FIXED BINARY (31,0),),
          RETCDE            FIXED BINARY (15,0),
          RFIELD            CHARACTER (27),
          SFIELD            CHARACTER (35),
        ;
        SFIELD = ' SORT FIELDS=(1,6,CH,A),SIZE=E9999 ';
        RFIELD = ' RECORD TYPE=F,LENGTH=(18) ';
        CALL IHESRTA (SFIELD,RFIELD,45056,RETCDE);
        PUT EDIT (RETCDE)  (SKIP(3),F(3));
        END SORT;


/* PRINT OUT THE SORTED RECORDS AS A CHECK.                            */


        ON ENDFILE (SORTOUT) GO TO CLOSE;
        OPEN FILE (SORTOUT);
RD:     READ FILE (SORTOUT) INTO (STR);
          PUT EDIT(STR)(X(5),A(18));
        GO TO RD;
CLOSE:  CLOSE FILE (SORTOUT);


/* READ IN ALL OUTPUT SPECIFICATIONS NEEDED TO PRINT THE INDEX.        */


        GET EDIT (PAGE,SIZE,WD,MARG,SZ,LINEH1,LINES1,LINEH2,LINET,
          LINES2,SPACE1,SPACE2,COL,(MG(I) DO I = 1 TO COL),NPB,NBH,NBR,
          LT,TITL,HEAD,QUE)
          (17 F(4),(COL) F(4),A(LT),A(13),A(1));


/* PRINTING OCCURS IN THIS BLOCK.                                      */
```

```


PRINT:  BEGIN;
        DECLARE
          COL#(COL,SIZE-LINES2-1) CHARACTER (SPACE1+15+SPACE2),
          I                 FIXED BINARY (31,0),
          IND               FIXED BINARY (31,0),
            ISSUE           CHARACTER (3),
          J                 FIXED BINARY (31,0),
          L                 FIXED BINARY (31,0),
          NN                FIXED BINARY (31,0),
          PG                CHARACTER (4),
          SKPH              FIXED BINARY (31,0),
          SKPP              FIXED BINARY (31,0),
          VT                CHARACTER (6) VARYING,
          YR                CHARACTER (2),
        ;

        ON ENDFILE (SORTOUT) GO TO STOP;
        OPEN FILE (OUTS) STREAM PRINT OUTPUT PAGESIZE(SIZE+4);
        OPEN FILE (SORTOUT);


/* MAKE INITIALIZATIONS TO BEGIN PRINTING.                            */


        IF QUE = 'R' THEN DO;
                              SKPP = SZ - 3;
                              SKPH = MARG;
                              END;
                     ELSE DO;
                              SKPP = MARG;
                              SKPH = SZ - 12;
                              END;
BLKCT:  DO I = 1 TO SIZE-LINES2-1;
        DO J = 1 TO COL;
        COL#(J,I) = ' ';
        END;
        END BLKOT;
        L = NBH;
        NN = 1;
        IND = 1;


/* PUT A HEADING ON THE TITLE PAGE:  PAGE NUMBER, 'AIM/ARM YR1-YR2', */
/* AND SUB-TITLE.                                                     */


        PUT FILE (OUTS) EDIT (PAGE,HEAD)
          (PAGE,LINE(LINEH1),COLUMN(SKPP),F(4),SKIP(0),COLUMN(SKPH),
          A(13));
        PUT FILE (OUTS) LINE(LINES1);
LP1:    DO I = 1 TO COL;
        PUT FILE (OUTS) EDIT ('VT NO','ISSUE','PAGE')
          (SKIP(0),COLUMN(MG(I)),A(5),X(SPACE1),A(5),X(SPACE2),A(4));
        END LP1;
        PUT FILE (OUTS) SKIP(1);


/* WE NOW FILL UP A COLUMN OF AN INDEX PAGE WITH INDEX DATA.  THE    */
/* COLUMN HAS A FEW LINES OF PRINT, THEN A BLANK LINE, A FEW MORE    */
/* LINES OF PRINT, A BLANK LINE, ETC.                                 */
```

```
LP2:    DO I = 1 TO L;
LP3:    DO J = 1 TO NPB;
        READ FILE (SORTOUT) INTO (STR);
        GET STRING (STR) EDIT (VT,PG,ISSUE,YR)
          (A(6),A(4),A(3),A(2));
        CALL DELETE;
        PUT STRING (COL#(IND,NN)) EDIT (VT,ISSUE,YR,PG)
          (X(6-WD),A(WD),X(SPACE1),A(3),A(2),X(SPACE2),A(4));
        NN = NN + 1;
        END LP3;
        NN = NN + 1;
        END LP2;


/* NOW WE MUST START A NEW COLUMN TO BE FILLED ON THIS PAGE.  SO      */
/* WE UP THE COLUMN SUBSCRIPT.  IF WE HAVE FILLED THE LAST COLUMN     */
/* ON THIS PAGE, CONTINUE HERE.                                       */


        IND = IND + 1;
        IF IND = (COL + 1)


                 /* ADD THE FINISHED PAGE TO THE OUTPUT   */
                 /* FILE.                                 */


          THEN DO;
               IND = 1;
               LP4: DO I = 1 TO L*NPB+L-1;
                    PUT FILE (OUTS) SKIP(1);
               LP5: DO J = 1 TO COL;
                    PUT FILE (OUTS) EDIT (COL#(J,I))
                      (SKIP(0),COLUMN(MG(J)-1),A(15+SPACE1+SPACE2));
                    COL#(J,I) = ' ';
                    END LP5;
                    END LP4;


                 /* PUT A NEW HEADING ON THE OUTPUT FILE  */
                 /* FOR THE NEW PAGE.                     */


               L = NBR;
               PAGE = PAGE + 1;
               IF SKPP = MARG THEN DO;
                                    SKPP = SZ - 3;
                                    SKPH = MARG;
                                    END;
                               ELSE DO;
                                    SKPP = MARG;
                                    SKPH = SZ - 12;
                                    END;
               PUT FILE (OUTS) EDIT (PAGE,HEAD,TITL)
                 (PAGE,LINE(LINEH2),COLUMN(SKPP),F(4),SKIP(0),COLUMN(
                 SKPH),A(13),LINE(LINET),X((SZ-MARG+1-LT)/2+MARG-1),
                 A(LT));
               PUT FILE (OUTS) LINE(LINES2);
               LP6: DO I = 1 TO COL;
                    PUT FILE (OUTS) EDIT ('VT NO','ISSUE','PAGE')
```

```
                     (SKIP(0),COLUMN(MG(I)),A(5),X(SPACE1),A(5),
                     X(SPACE2),A(4));
                    END LP6;
               PUT FILE (OUTS) SKIP(1);
               END;


/* NEW PAGE OR NOT, INITIALIZE THE LINE COUNTER AND CONTINUE            */
/* PROCESSING THE INDEX DATA.                                           */


        NN = 1;
        GO TO LP2;


/* THIS PROCEDURE DELETES LEADING ZEROES FROM A VT NUMBER.  THUS,       */
/* '007095' BECOMES '7095'.                                             */


DELETE: PROCEDURE;
          DECLARE
          M                  FIXED BINARY (31,0),
          TEMP               CHARACTER (6) VARYING,
        ;
        TEMP = VT;
LP8:    DO M = 1 TO 6;
        IF SUBSTR(TEMP,M,1) = '0'
          THEN DO;
               IF M = 1 THEN VT = ' ' || SUBSTR(VT,2);
                        ELSE VT = SUBSTR(VT,1,M-1) || ' ' || SUBSTR(VT,
                                                               M+1);
               END;
               ELSE GO TO FIX;
        END LP8;
FIX:    VT = SUBSTR(VT,6-WD+1,WD);
        RETURN;
        END DELETE;


/* WHEN ALL THE DATA HAS BEEN READ OFF THE DISK FILE, WE ARE ALMOST     */
/* THROUGH.  ADD THE PAGE THAT WAS BEING FILLED TO THE OUTPUT FILE.     */
/* CLOSE ALL FILES AND TERMINATE THE PROGRAM.  THE INDEX IS COMPLETE.*/


STOP:     CLOSE FILE (SORTOUT);
          LP7: DO I = 1 TO L*NPB+L-1;
               PUT FILE (OUTS) SKIP(1);
          LP8: DO J = 1 TO COL;
               PUT FILE (OUTS) EDIT (COL#(J,I))
                 (SKIP(0),COLUMN(MG(J)-1),A(15+SPACE1+SPACE2));
               END LP8;
               END LP7;
          PUT FILE (OUTS) PAGE;
          CLOSE FILE (OUTS);
          PUT EDIT ('LAST PAGE USED IS ',PAGE) (SKIP(3),A(18),F(4));

        END PRINT;

        END ANNDOC;
```

EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICES

```
/**********************************************************************/
/*                                                                    */
/*    E D I T   F O R   S U B J E C T ,   A U T H O R ,   A N D       */
/*              S . S .    S U B S E T    F I L E S                   */
/*                                                                    */
/*                                                                    */
/*  PURPOSE:  THIS PROGRAM CAN COPY THE EIGHT SUBJECT DATA SETS ON    */
/*            MAGNETIC TAPE TO ANOTHER TAPE.  RECORDS FROM EACH DATA  */
/*            SET CAN BE ADDED, DELETED, OR REPLACED WHILE THE        */
/*            COPYING IS BEING DONE                                   */
/*                                                                    */
/*INPUT TO THE PROGRAM:                                               */
/*    -SYSIN-, INPUT ON CARDS:                                        */
/*      THE FOLLOWING CARD SET-UP IS REPEATED FOR EACH DATA SET ON    */
/*      THE TAPE.  FORMAT ON THESE CARDS IS  'FREE'; THE DATA CAN BE  */
/*      IN ANY COLUMNS AS LONG AS AT LEAST ONE BLANK IS BETWEEN       */
/*      VARIABLES.  THE ORDER OF THE DATA IS AS FOLLOWS:              */
/*      -M-, THE NUMBER OF RECORDS TO BE REPLACED ON THE DATA SET.    */
/*      -K-, THE NUMBER OF RECORDS TO BE DELETED FROM THE DATA SET.   */
/*      'REPLACEMENT DECK' -- THIS IS A DECK WITH THE DATA TO BE      */
/*        REPLACED AND ITS REPLACEMENT.  FIRST, WE HAVE A KEYWORD,    */
/*        A DATE, AND A VT NUMBER, EACH IN QUOTES.  THIS IS DATA IN   */
/*        A RECORD WHICH IS TO BE REPLACED IN SOME PART.  THEN, WE    */
/*        HAVE THE KEYWORD, DATE, AND VT NUMBER FOR THE REPLACEMENT.  */
/*        THIS PATTERN CONTINUES 'M' TIMES.                           */
/*      'DELETION DECK' -- THIS GIVES THE KEYWORD, DATE, AND VT       */
/*        NUMBER, EACH IN QUOTES, OF A RECORD TO BE DELETED.  THIS    */
/*        OCCURRS -K- TIMES.                                          */
/*      -L-, THE NUMBER OF RECORDS TO BE ADDED TO THE DATA SET.       */
/*      'ADDITION DECK' -- THIS GIVES THE KEYWORD, DATE, AND VT       */
/*        NUMBER, EACH IN QUOTES, OF A RECORD TO BE ADDED.  THIS      */
/*        OCCURRS -L- TIMES.                                          */
/*            NOTE:  IF L=0, THERE ARE NO ADDITIONS.  IF K=0, THERE   */
/*                   ARE NO DELETIONS.  IF M=0, THERE ARE NO          */
/*                   REPLACEMENTS.  IF ALL THREE ARE ZERO, THE DATA   */
/*                   SET WILL ONLY BE COPIED.                         */
/*                                                                    */
/**********************************************************************/

EDIT:   PROCEDURE OPTIONS (MAIN);
DECLARE (SUB_LEN,RECORD_LEN) FIXED BINARY(31,0),
        INDEX CHARACTER(4);
        GET FILE(SYSIN) LIST(SUB_LEN,INDEX);
        RECORD_LEN=SUB_LEN+8;
    IF RECORD_LEN<18 THEN RECORD_LEN=18;
BEGIN;
        DECLARE
          FIX                   FILE SEQUENTIAL RECORD,
          TEMP                  FILE SEQUENTIAL RECORD,
          BLANK                 CHARACTER (1),
          DUMMY                 CHARACTER (RECORD_LEN),
          REC(50)               CHARACTER (RECORD_LEN),
          VT                    CHARACTER (6),
          PUB                   CHARACTER(2),
          SUB                   CHARACTER (SUB_LEN) VARYING,
          END                   CHARACTER (1) INITIAL ('"'),
          SUBJ                  CHARACTER (SUB_LEN) VARYING,
          DUM(100,2)            CHARACTER (RECORD_LEN),
        ;
 GET FILE(SYSIN) LIST (M,K);
        IF M = 0 THEN GO TO TEST;
LOOP3:  DO I = 1 TO M;
```

```
LOOP3A: DO II = 1 TO 2;
         GET LIST (SUB,PUB,VT);
                    IF INDEX='SUBJ' THEN SUBJ=SUB||END;
                                     ELSE SUBJ=SUB;
 DUM(I,II)=' ';
PUT STRING(DUM(I,II)) EDIT(SUBJ,PUB,VT)(A(SUB_LEN),A(2),A(6));
         END LOOP3A;
         END LOOP3;
TEST:    IF K = 0 THEN GO TO OPEN;
LOOP3B: DO I = 1 TO K;
         GET LIST (SUB,PUB,VT);
                    IF INDEX='SUBJ' THEN SUBJ=SUB||END;
                                     ELSE SUBJ=SUB;
    REC(I)=' ';
PUT STRING(REC(I)    ) EDIT(SUBJ,PUB,VT)(A(SUB_LEN),A(2),A(6));
         END LOOP3B;
OPEN: OPEN FILE(FIX) INPUT SEQUENTIAL RECORD;
       OPEN FILE(TEMP) OUTPUT SEQUENTIAL RECORD;
         ON ENDFILE (FIX) BEGIN;
                    GET LIST (L);
           PUT FILE (SYSPRINT) SKIP(4);
           IF L = 0 THEN GO TO CL;
           LOOP5: DO I = 1 TO L;
                   GET LIST (SUB,PUB,VT);
                   SUBJ = SUB || END;
    DUMMY=' ';
PUT STRING(DUMMY     ) EDIT(SUBJ,PUB,VT)(A(SUB_LEN),A(2),A(6));
                   WRITE FILE (TEMP) FROM (DUMMY);
                   PUT DATA (DUMMY);
                   END LOOP5;
 CL:    CLOSE FILE(FIX), FILE(TEMP);
           GO TO EL2;
           END;
NXT:     READ FILE (FIX) INTO (DUMMY);
         IF M = 0 THEN GO TO LBL1;
LOOP4:   DO I = 1 TO M;
         IF DUMMY = DUM(I,1) THEN DO;
                                     DUMMY = DUM(I,2);
                                     WRITE FILE (TEMP) FROM (DUMMY);
                                     PUT DATA (DUM(I,2));
                                     DUM(I,2) = 'REPLACED';
                                     GO TO NXT;
                                     END;
         END LOOP4;
LBL1:    IF K = 0 THEN GO TO LBL2;
LOOP4A: DO I = 1 TO K;
         IF DUMMY = REC(I) THEN DO;
                                   REC(I) = 'DELETED';
                                   PUT LIST ('DELETED');
                                   GO TO NXT;
                                   END;
         END LOOP4A;
 LBL2:   WRITE FILE (TEMP) FROM (DUMMY);
         GO TO NXT;
 EL2:
     ;
   STOP:    END EDIT;
```

DOCUMENT NUMBER EDIT

```
VTISSUE: PROCEDURE OPTIONS (MAIN);                                        001010
     DCL            1 BUFFER_DISC,                                        001020
                    2 DISC_KEY CHAR (6),                                  001030
                    2 DISC_INFO  CHAR (12),                               001040
                    1 CARD_IN,                                            001050
                    2 CARD_KEY CHAR (6),                                  001070
                    2 ACTUAL_INFO CHAR (12),                              001080
                    2 FILL CHAR (61),                                     001090
                    2 CHECK_1  CHAR (1);                                  001100
     DCL            END_CDE FIXED BINARY (15,0), CARD_HOLD CHAR (18);     001110
                    END_CDE=0;                                            001120
                    ON ENDFILE (DISC) GO TO EOJ2;                         001130
                    READ FILE (DISC) INTO (BUFFER_DISC);                  001140
     READ1:         CARD_HOLD=' ';                                        001150
                    ON ENDFILE (CARDS) GO TO EOJ1;                        001160
                    READ FILE (CARDS) INTO (CARD_IN);                     001170
                    CARD_HOLD=CARD_KEY||ACTUAL_INFO;                      001180
                    IF CHECK_1= 'A' THEN GO TO ADDT;                      001190
     DUPS:          IF CARD_KEY = DISC_KEY THEN GO TO READ_DISC;          001200
                    IF CARD_KEY > DISC_KEY THEN                           001210
                    WRITE FILE (TAPE) FROM (BUFFER_DISC);                 001220
                    IF CARD_KEY < DISC_KEY THEN GO TO READ1;              001230
     READ_DISC: READ FILE (DISC) INTO (BUFFER_DISC);                      001240
                    GO TO DUPS;                                           001250
     ADDT:          IF CARD_KEY > DISC_KEY THEN DO;                       002010
                    WRITE FILE (TAPE) FROM (CARD_HOLD);                   002020
                    GO TO READ1;                                          002030
                       END;                                               002031
                    ELSE WRITE FILE (TAPE) FROM (BUFFER_DISC);            002040
                    READ FILE (DISC) INTO (BUFFER_DISC);                  002050
                    GO TO ADDT;                                           002060
     EOJ1:          END_CDE=1;                                            002070
                    READ FILE (DISC) INTO (BUFFER_DISC);                  002080
                    WRITE FILE (TAPE)FROM (BUFFER_DISC);                  002090
                    GO TO EOJ1;                                           002100
     EOJ2:          IF END_CDE = 1 THEN GO TO CLOSE_FILES;                002110
                    ON ENDFILE (CARDS) GO TO CLOSE_FILES;                 002120
                    CARD_HOLD=' ';                                        002130
                    READ FILE (CARDS) INTO (CARD_IN);                     002140
                    CARD_HOLD=CARD_KEY||ACTUAL_INFO;                      002150
                    WRITE FILE (TAPE) FROM (CARD_HOLD);                   002160
                    GO TO EOJ2;                                           002170
     CLOSE_FILES: CLOSE FILE (CARDS), FILE (DISC), FILE (TAPE);           002180
                    END;                                                  002190
```

RIE FORMAT

```
/*
```

```
        /*
        PURPOSE: REFORMATING OF BIBLIOGRAPHIC AND ABSTRACT TAPES INTO
                 RIE FORMAT, THUS ALLOWING QUERY TO SEARCH THE DATA BASE.
        */
 CRIE: PROCEDURE OPTIONS(MAIN);
 DCL
      CODE_TABLE(20) FIXED BINARY(15,0), /* WILL BE TREATED AS BIT */
      BUFFER CHAR(6996) VAR,             /* USED AS I/O WORK AREA  */
     (LENGTHS,LEN_P_AUTHOR(2),#_P_AUTHORS,TITLE_LEN,#_C_AUTHORS,
       LEN_C_AUTHOR(5),#SA,LEN_SA(3),LEN_PRICE,LEN_CONTRACT,LEN_GRANT,
       LEN_PA,LEN_REPORT,LEN_AVAIL,LEN_JC,LEN_DATE,LEN_PAGES,LEN_DESC,
         LEN_DESC_NOTE,LEN_IDENT) FIXED BINARY(15,0),
     VT CHAR(10) VAR,
     ED CHAR(20) VAR,
      PERSONAL_AUTHOR CHAR(264) VAR,
      TITLE           CHAR(396) VAR,
      CORPORATE_AUTHOR CHAR(660) VAR,
      SPONSOR_AGENCY  CHAR(396)  VAR,
      PA              CHAR(2),
      (PRICE,CONTRACT,GRANT,REPORT,BUREAU) CHAR(66) VAR,
      DATE            CHAR(7),
      PAGES           CHAR(16)  VAR,
     (DESC_NOTE,JOURNAL) CHAR(132) VAR,
      AVAIL           CHAR(264) VAR,
     DESCRIPTOR       CHAR(2000) VAR,
     EOF FIXED BINARY(15,0),
     ISSUE CHAR(15) VAR,
     ISSUEQ CHAR(2),
      IDENTIFIER      CHAR(396) VAR,
      BLANKS          CHAR(660) INITIAL((660)' '),
     CODE_CHAR(20) CHAR(2) DEFINED CODE_TABLE,
     PA_LEN CHAR(2) DEFINED LEN_PA,
     LENGTHS_CHAR CHAR(2) DEFINED LENGTHS,
     DATE_LEN CHAR(2) DEFINED LEN_DATE,
     I_CHAR CHAR(2) DEFINED I,
     LEN_CHAR CHAR(2) DEFINED LEN_DESC,
      SHORT_TITLE     CHAR(396) VAR;

  /*
           E O F CONTROL
  */

 ON ENDFILE(RIE) GO TO STOP;

  /*
           SET UP C O D E  T A B L E
  */

 CODE_TABLE(1)=16; CODE_TABLE(2)=17; CODE_TABLE(4)=23; CODE_TABLE(5)=26;
 CODE_TABLE(6)=27; CODE_TABLE(19)=128; CODE_TABLE(20)=132;
 DO I=7 TO 10; CODE_TABLE(I)=28+I; END;
 DO I=11 TO 18;  CODE_TABLE(I)=32+I; END; CODE_TABLE(3)=20;
        OPEN FILE(ABSTRACT) SEQUENTIAL BUFFERED RECORD, INPUT;

  /*
           READ FILE  R I E  INTO BUFFER -- WORK AREA
  */

```

```
/*


   INPUT:
 READ FILE(RIE) INTO(BUFFER); I=INDEX(BUFFER,'||')+1;
 LEN_P_AUTHOR=0; LEN_C_AUTHOR=0; LEN_SA=0;

   /*
            DETERMINE WHETHER RECORD IS AIM OR ARM ISSUE
   */

      IF SUBSTR(BUFFER,49,1)='A' THEN ISSUE='AIM ';
                                   ELSE ISSUE='ARM ';

   /*
            DETERMINE QUARTER OF ISSUE
   */

      ISSUEQ=SUBSTR(BUFFER,49,2);
  IF ISSUEQ='F ' THEN ISSUE=ISSUE||'FALL'||SUBSTR(BUFFER,51,4);
                 ELSE
  IF ISSUEQ='W ' THEN ISSUE=ISSUE||'WINTER'||SUBSTR(BUFFER,51,4);
                 ELSE
  IF ISSUEQ='SP' THEN ISSUE=ISSUE||'SPRING'||SUBSTR(BUFFER,51,4);
                 ELSE
  IF ISSUEQ='SU' THEN ISSUE=ISSUE||'SUMMER'||SUBSTR(BUFFER,51,4);

   /*
            GET STRING IN BUFFER AND EDIT INFO ON IT
   */

   GET STRING(BUFFER) EDIT(#_P_AUTHORS,(LEN_P_AUTHOR(J) DO J=1 TO #_P_AU
  THORS),TITLE_LEN,#_C_AUTHORS,(LEN_C_AUTHOR(J) DO J=1 TO #_C_AUTHORS),
  #SA,(LEN_SA(J) DO J=1 TO #SA),LEN_PRICE,LEN_CONTRACT,LEN_GRANT,LEN_PA,
  LEN_REPORT,LEN_BUREAU,LEN_AVAIL,LEN_JC,LEN_DATE,LEN_PAGES,LEN_DESC_NOTE
  ,LEN_DESC,LEN_IDENT)(X(I),28 F(4));

   /*
            GET VT AND ED NUMBERS
   */

  GET STRING(BUFFER) EDIT(VT,ED)(A(10),X(17),A(20));
       J=58;
       K=SUM(LEN_P_AUTHOR);

   /*
            CHECK IF THERE ARE PERSONAL AUTHORS
                  IF SO, GET P. A. AND TITLE
                  IF NOT, GET TITLE
   */

       IF K>0 THEN
  GET STRING(BUFFER) EDIT(PERSONAL_AUTHOR,TITLE)(X(J),A(K),A(TITLE_LEN));
                      ELSE
  GET STRING(BUFFER) EDIT(TITLE)(X(J),A(TITLE_LEN));
       J=J+K+TITLE_LEN;
       K=SUM(LEN_C_AUTHOR);

   /*
```

```
/*
            CHECK IF THERE ARE CORPORATE AUTHORS
                 IF SO, GET C.A.
   */

      IF K>0 THEN DO;
  GET STRING(BUFFER) EDIT(CORPORATE_AUTHOR)(X(J),A(K));
       J=J+K;
                 END;
       K=SUM(LEN_SA);

   /*
            CHECK IF THERE ARE SPONSORING AGENCIES
                 IF SO, GET S.A.
   */

      IF K>0 THEN DO;
  GET STRING(BUFFER) EDIT(SPONSOR_AGENCY)(X(J),A(K));
       J=J+K;
                 END;

   /*
            GET STRING IN BUFFER AND EDIT MORE INFORMATION ON FILE
   */

  GET STRING(BUFFER) EDIT(
               PRICE,CONTRACT,GRANT,PA,REPORT,BUREAU,AVAIL,JOURNAL,DATE
  ,PAGES,DESC_NOTE,DESCRIPTOR,IDENTIFIER)(X(J),A(LEN_PRICE),
        A(LEN_CONTRACT),A(LEN_GRANT),A(LEN_PA),A(LEN_REPORT),A(LEN_BUREAU
  ),A(LEN_AVAIL),A(LEN_JC),A(LEN_DATE),A(LEN_PAGES),A(LEN_DESC_NOTE),
  A(LEN_DESC),A(LEN_IDENT));

   /*
            CHECK IF THERE IS AN ASSIGNED ED NUMBER
                 IF SO, GO TO R I E INPUT
                 IF NOT, CONTINUE
   */

      IF SUBSTR(ED,3,1)='('|SUBSTR(ED,11,1)¬=' ' THEN DO;
               GO TO INPUT;
                                                    END;
   LENGTHS=12;
    ED=SUBSTR(ED,1,2)||SUBSTR(ED,4,3)||SUBSTR(ED,8,3);
    VT=SUBSTR(VT,1,2)||SUBSTR(VT,4,3)||SUBSTR(VT,8,3);

   /*
            CHECK IF THERE IS AN ED NUMBER
                 IF SO, PLACE VT AND ED CODES, LENGTHS, AND NUMBERS
                 INTO BUFFER
                 IF NOT, PLACE VT C, L, AND N INTO BUFFER
   */

   IF SUBSTR(ED,4,1)¬=' ' THEN DO;
      BUFFER=LENGTHS_CHAR||CODE_CHAR(1)||ED;
      BUFFER=BUFFER||LENGTHS_CHAR||CODE_CHAR(2)||VT;
      END;
    ELSE
```

```
/*

 BUFFER=LENGTHS_CHAR||CODE_CHAR(2)||VT;

 /*
          CHECK IF THERE IS A PROGRAM AREA
               IF SO, PUT P.A. C, L, AND N INTO BUFFER
 */

 IF LEN_PA¬=0 THEN DO;
    LEN_PA=LEN_PA+4;
    BUFFER=BUFFER||PA_LEN||CODE_CHAR(3)||PA;
  END;

 /*
          CHECK IF THERE IS A DATE
               IF SO, PUT DATE CODE, LENGTH, AND DATA INTO BUFFER
 */

 IF DATE¬='        '&DATE¬='        ' THEN DO;
    LEN_DATE=LEN_DATE+4;
    UNSPEC(DATE)=UNSPEC(DATE)|UNSPEC(SUBSTR(BLANKS,1,LEN_DATE-4));
    BUFFER=BUFFER||DATE_LEN||CODE_CHAR(4)||DATE;
                                         END;

 /*
          CHECK IF THERE IS A TITLE
               IF SO, PUT TITLE C, L, AND DATA INTO BUFFER
 */

 IF TITLE_LEN¬=0 THEN DO;
    UNSPEC(TITLE)=UNSPEC(TITLE)|UNSPEC(SUBSTR(BLANKS,1,TITLE_LEN));
BTITLE: I=INDEX(TITLE,'<');
        IF I=0 THEN GO TO ENDTITLE;
               ELSE IF I=1 THEN TITLE=SUBSTR(TITLE,2);
                           ELSE TITLE=SUBSTR(TITLE,1,I-1)||SUBSTR(TITLE
                                ,I+1);
        I=INDEX(TITLE,'>');
        IF I=0 THEN DO;
                    PUT FILE(SYSPRINT) LIST('** TITLE ERROR**',VT);
                    GO TO INPUT;
                    END;
               ELSE TITLE=SUBSTR(TITLE, 1,I-1)||SUBSTR(TITLE,I+2);
        GO TO BTITLE;
ENDTITLE: I=LENGTH(TITLE)+4;
    BUFFER=BUFFER||I_CHAR||CODE_CHAR(5)||TITLE;
                      END;

 /*
          CHECK IF THERE IS A PERSONAL AUTHOR
               IF SO, PUT P. A.  C, L, AND DATA INTO BUFFER
 */

 IF #_P_AUTHORS>0 THEN DO;
    I=SUM(LEN_P_AUTHOR);
    UNSPEC(PERSONAL_AUTHOR)=UNSPEC(PERSONAL_AUTHOR)|UNSPEC(SUBSTR(
         BLANKS,1,I));
    IF #_P_AUTHORS=2 THEN DO;
```

```
/*

           SUBSTR(PERSONAL_AUTHOR, LEN_P_AUTHOR(I)+1)=';'||SUBSTR(
                PERSONAL_AUTHOR,LEN_P_AUTHOR(I)+1);
           I=I+1;
                            END;
      I=I+4;
      BUFFER=BUFFER||I_CHAR||CODE_CHAR(6)||PERSONAL_AUTHOR;
                         END;

   /*
            BLOCK FOR MANIPULATING DESCRIPTOR CODE, LENGTH, AND DATA
   */

     LEN_DESC_NOTE=LEN_DESC_NOTE+LEN_PAGES;
     DESC_NOTE=PAGES||DESC_NOTE;
  DO I=7 TO 18;
     IF I=8 THEN DO; LEN_DESC=LEN_IDENT;
                      DESCRIPTOR=IDENTIFIER;
                  END;
             ELSE
     IF I=9 THEN DO; LEN_DESC=LEN_PRICE;
                      DESCRIPTOR=PRICE;
                   END;
             ELSE
     IF I=10 THEN DO; LEN_DESC=LEN_DESC_NOTE;
                       DESCRIPTOR=DESC_NOTE;
                   END;
             ELSE
       IF I=11 THEN DO; LEN_DESC=LENGTH(ISSUE);
                        DESCRIPTOR=ISSUE;
                    END;
           ELSE
       IF I=12 THEN DO;
          EOF=1;
       ON ENDFILE(ABSTRACT) BEGIN;
         PUT FILE(SYSPRINT) EDIT(EOF,VT)(SKIP,F(4),A(10));
         IF EOF=2 THEN GO TO LOOPEND;
            EOF=2;
         CLOSE FILE(ABSTRACT);
           OPEN FILE(ABSTRACT) SEQUENTIAL BUFFERED RECORD  INPUT;
            GO TO ABSL;
                            END;
         GO TO ABVT;
                   END;
       ELSE
      IF I=13 THEN DO; LEN_DESC=LEN_REPORT;
                       DESCRIPTOR=REPORT;
                   END;
              ELSE
       IF I=14 THEN DO; LEN_DESC=LEN_CONTRACT;
                       DESCRIPTOR=CONTRACT;
                    END;
              ELSE
      IF I=15 THEN DO; LEN_DESC=LEN_GRANT;
                       DESCRIPTOR=GRANT;
                   END;
              ELSE
```

```
/*


         IF I=16 THEN DO; LEN_DESC=LEN_BUREAU;
                                 DESCRIPTOR=BUREAU;
                           END;
                    ELSE
         IF I=17 THEN DO; LEN_DESC=LEN_AVAIL;
                                 DESCRIPTOR=AVAIL;
                           END;
         IF I=18 THEN DO; LEN_DESC=LEN_JC;
                                 DESCRIPTOR=JOURNAL;
                           END;
                  IF LEN_DESC¬=0 THEN DO;
                   UNSPEC(DESCRIPTOR)=UNSPEC(DESCRIPTOR)||UNSPEC(SUBSTR(
                         BLANKS,1,LEN_DESC));
     INSERT: LEN_DESC=LEN_DESC+4;
           BUFFER=BUFFER||LEN_CHAR||CODE_CHAR(I)||DESCRIPTOR;
                         LEN_DESC=0;
           GO TO LOOPEND;

      /*
                   END OF DESCRIPTOR BLOCK

      /*
                   CONSTRUCT ABSTRACT VT NUMBER
      */

     ABVT: VT='VT '||SUBSTR(VT,3,3)||' '||SUBSTR(VT,6,3);

      /*
                   READ THE FILE ABSTRACT INTO DESCRIPTOR -- WORK AREA
      */

     ABSL: READ FILE(ABSTRACT) INTO(DESCRIPTOR);
      IF SUBSTR(DESCRIPTOR,1,10)¬=VT THEN GO TO ABSL;

      /*
                   SPECIFY DESCRIPTOR DATA AND LENGTH OF DATA
      */

                         DESCRIPTOR=SUBSTR(DESCRIPTOR,13);
                         LEN_DESC=LENGTH(DESCRIPTOR);
              GO TO INSERT;
     END;
     LOOPEND: END;

       /*
                    CHECK IF THERE ARE ANY CORPORATE AUTHORS
                           IF SO, PUT C.A. CODE, LENGTH, AND DATA INTO BUFFER
       */

        IF #_C_AUTHORS>0 THEN DO;
           I=SUM(LEN_C_AUTHOR); J=0;
           UNSPEC(CORPORATE_AUTHOR)=UNSPEC(CORPORATE_AUTHOR)||UNSPEC(SUBSTR(
                  BLANKS,1,I));
           DESCRIPTOR=CORPORATE_AUTHOR;
            IF #_C_AUTHORS>1 THEN DO K=2 TO #_C_AUTHORS;
                 J=J+LEN_C_AUTHOR(K-1)+1;
```

```
/*

         SUBSTR(DESCRIPTOR,J,LEN_C_AUTHOR(K)+1)=';'||SUBSTR(
              CORPORATE_AUTHOR,LEN_C_AUTHOR(K-1),LEN_C_AUTHOR(K));
                             END;
    I=I+#_C_AUTHORS+3;
    BUFFER=BUFFER||I_CHAR||CODE_CHAR(19)||DESCRIPTOR;
                           END;

 /*
         CHECK IF THERE ARE ANY SPONSORING AGENCIES
              IF SO, PUT S.A. CODE, LENGTH, AND DATA INTO BUFFER
 */

IF #SA>0 THEN DO;
    DESCRIPTOR=SPONSOR_AGENCY;
    I=SUM(LEN_SA);
    UNSPEC(SPONSOR_AGENCY)=UNSPEC(SPONSOR_AGENCY)|UNSPEC(SUBSTR(BLANKS,
      1,I));   J=0;
   IF #SA>1 THEN DO K=2 TO #SA;
         J=J+LEN_SA(K-1)+1;
         SUBSTR(DESCRIPTOR,J,LEN_SA(K)+1)=';'||SUBSTR(
              SPONSOR_AGENCY,J-K+1,LEN_SA(K));
                    END;
    I=I+#SA+3;
    BUFFER=BUFFER||I_CHAR||CODE_CHAR(20)||DESCRIPTOR;
              END;

 /*
         WRITE FILE BUFFER ONTO THE MASTER FILE
 */

WRITE FILE(MASTER) FROM(BUFFER);
GO TO INPUT;
STOP: END CRIE;
```

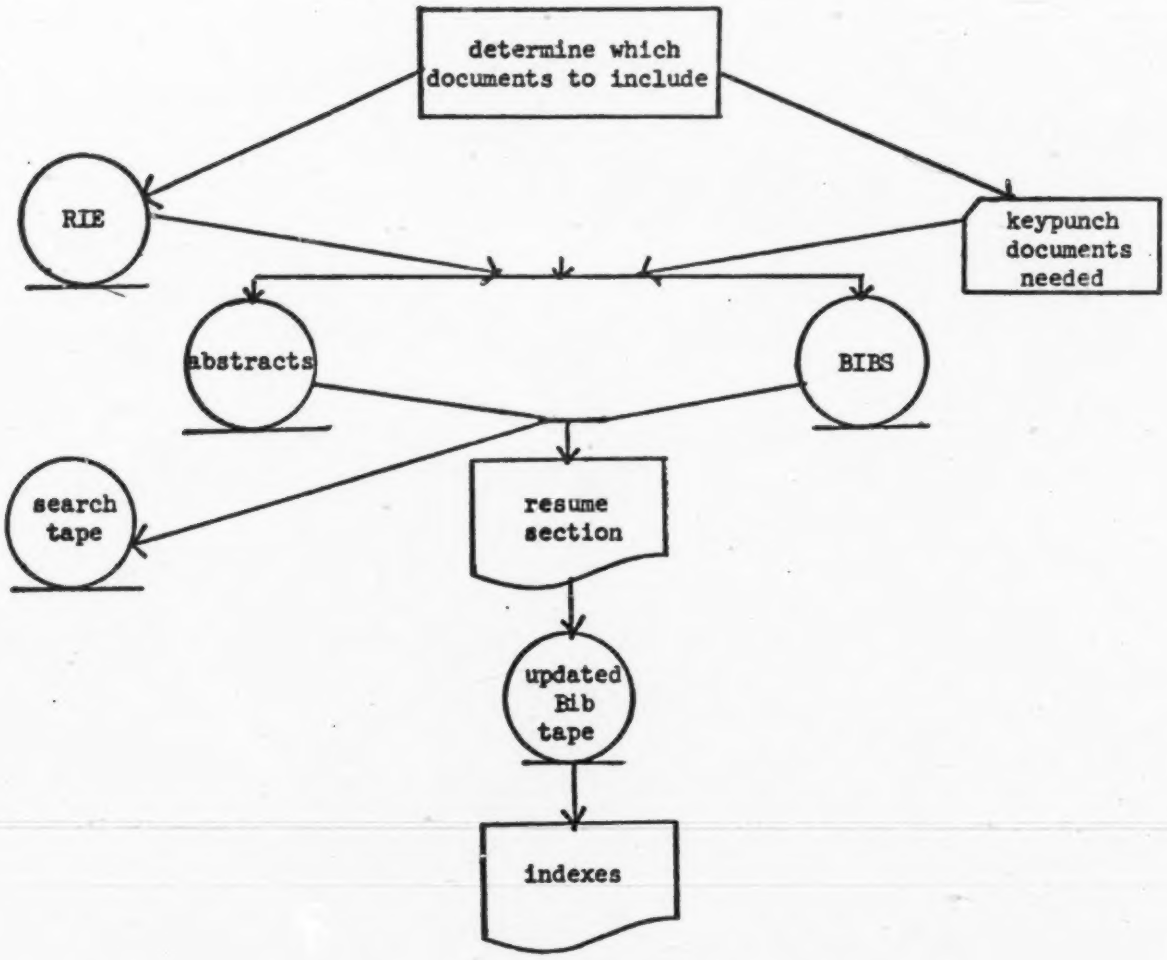
APPENDIX F

PROGRAM AND SYSTEM FLOWCHARTS

	page
OVERVIEW OF SYSTEM	211
DETAILED FLOWCHART OF SYSTEM	213
RIE TO AIM/ARM	218
MASTER	221
ABSTRACT	231
RESUME PRINT	234
MASTER EDIT	236
SUBJECT QUARTERLY	238
AUTHOR QUARTERLY	240
SUPPORTING SERVICES QUARTERLY	242
DOCUMENT NUMBER QUARTERLY	244
CONVERSION OF DOCUMENT NUMBER	246
SUBJECT ANNUAL	248
AUTHOR ANNUAL	250
SUPPORTING SERVICES ANNUAL	252
DOCUMENT NUMBER ANNUAL	254
EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICES	256
DOCUMENT NUMBER EDIT	258
RIE FORMAT	261

OVERVIEW OF SYSTEM

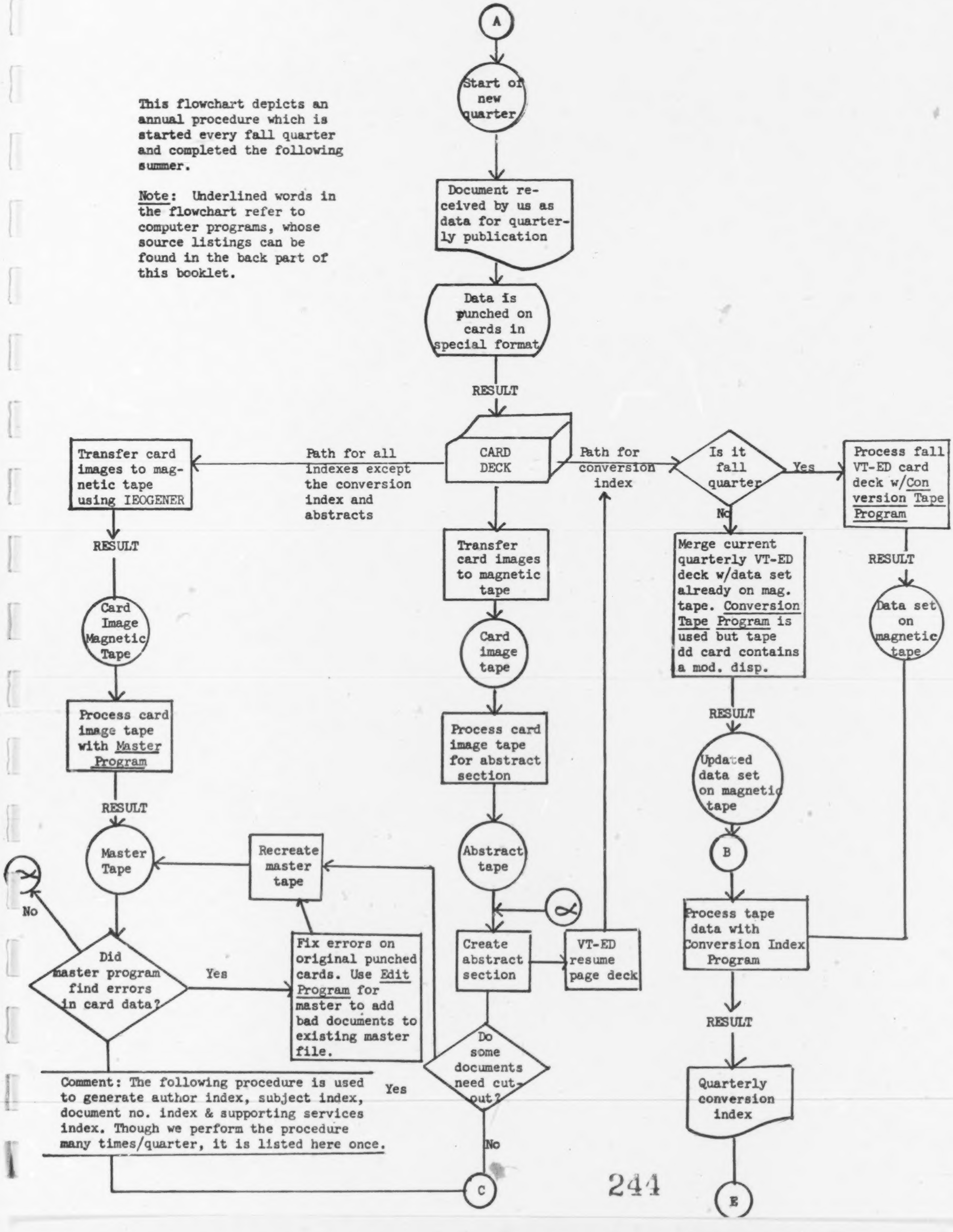
OVERVIEW OF THE SYSTEM


determine which documents to include
RIE
keypunch documents needed
abstracts
BIBS
search tape
resume section
updated Bib tape
indexes

DETAILED FLOWCHART OF SYSTEM

Flowchart of the System

(The System Is the Same for AIM and ARM)

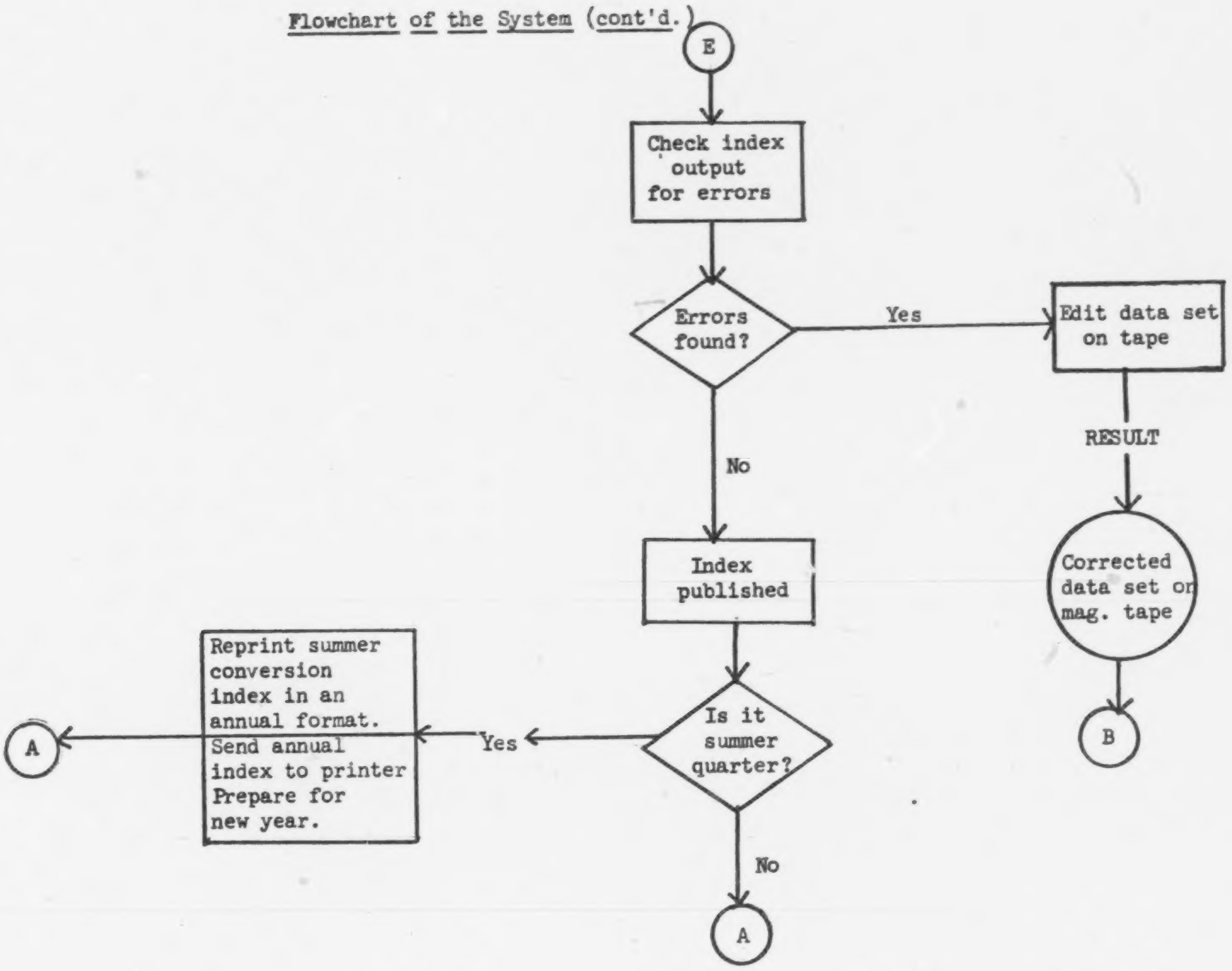

A
Start of new quarter
Document received by us as data for quarterly publication
Data is punched on cards in special format
RESULT
CARD DECK
Path for all indexes except the conversion index and abstracts
Transfer card images to magnetic tape using IEOGENER
RESULT
Card Image Magnetic Tape
Process card image tape with Master Program
RESULT
Master Tape
Did master program find errors in card data?
No
Yes
Fix errors on original punched cards. Use Edit Program for master to add bad documents to existing master file.
Recreate master tape
Transfer card images to magnetic tape
Card image tape
Process card image tape for abstract section
Abstract tape
Create abstract section
VT-ED resume page deck
Do some documents need cut-out?
Yes
No
C
Path for conversion index
Is it fall quarter
Yes
No
Process fall VT-ED card deck w/Conversion Tape Program
RESULT
Data set on magnetic tape
Merge current quarterly VT-ED deck w/data set already on mag. tape. Conversion Tape Program is used but tape dd card contains a mod. disp.
RESULT
Updated data set on magnetic tape
B
Process tape data with Conversion Index Program
RESULT
Quarterly conversion index
E

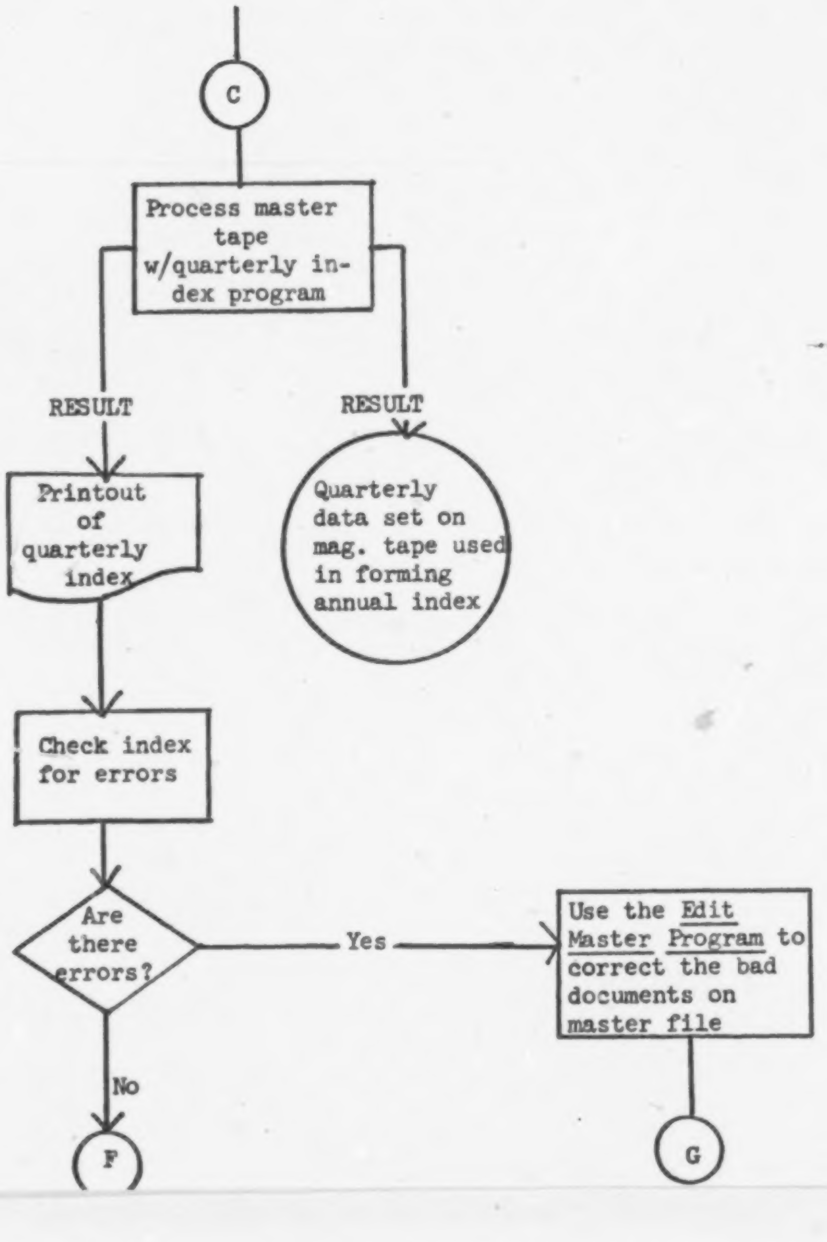
This flowchart depicts an annual procedure which is started every fall quarter and completed the following summer.

Note: Underlined words in the flowchart refer to computer programs, whose source listings can be found in the back part of this booklet.

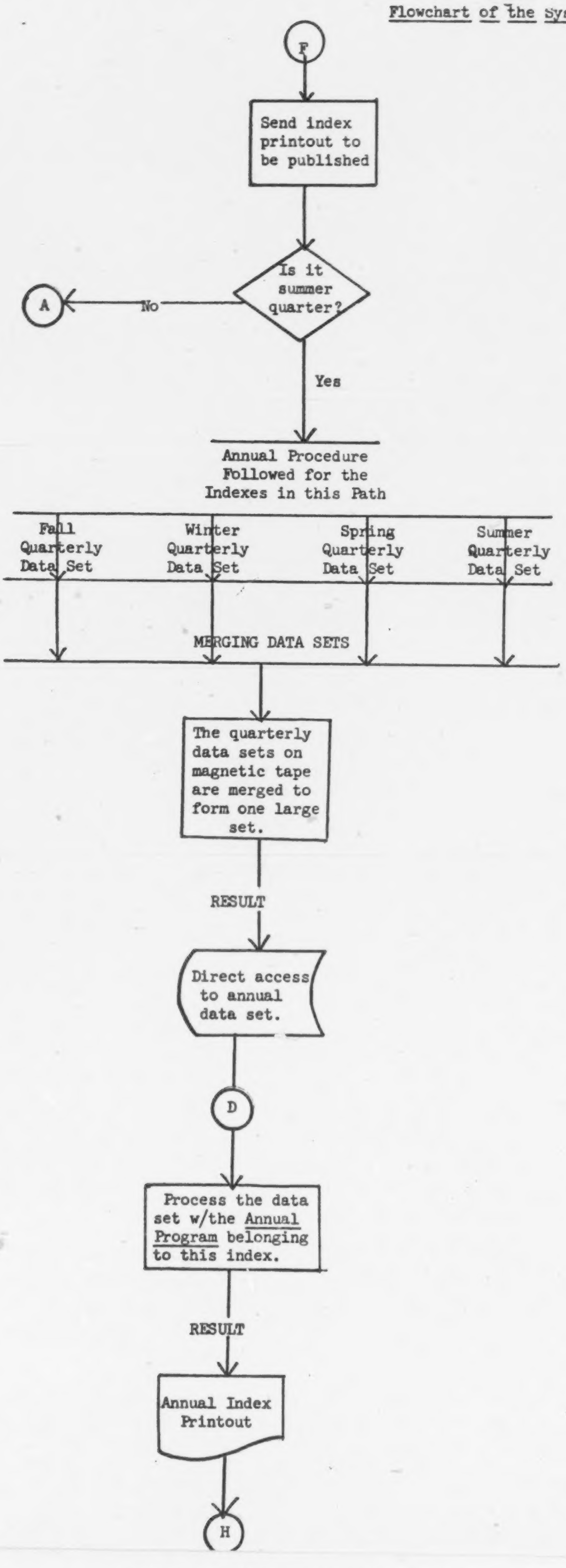
Comment: The following procedure is used to generate author index, subject index, document no. index & supporting services index. Though we perform the procedure many times/quarter, it is listed here once.

Flowchart of the System (cont'd.)

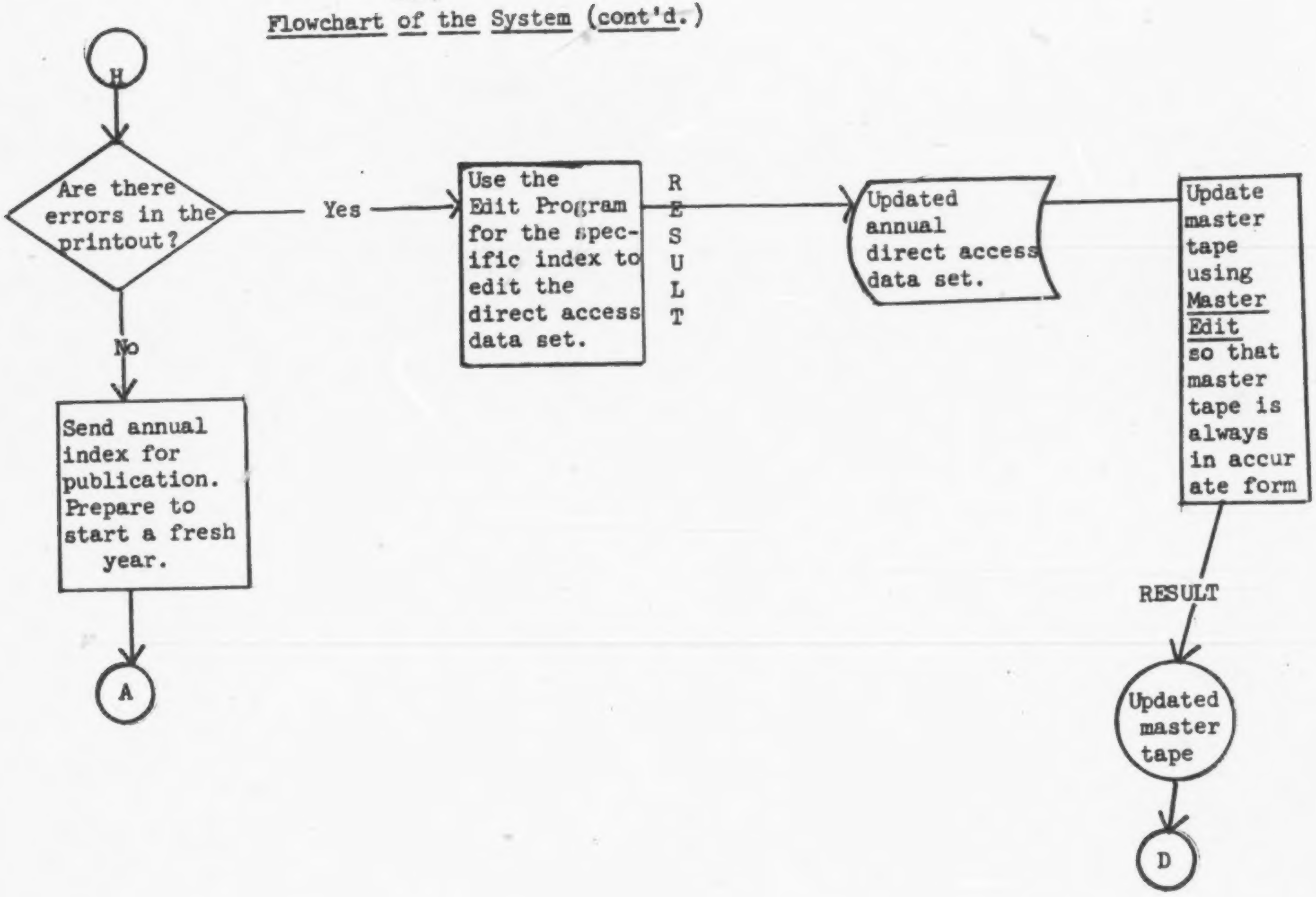

E
Check index output for errors
Errors found?
Yes
Edit data set on tape
RESULT
Corrected data set on mag. tape
B
No
Index published
Is it summer quarter?
Yes
Reprint summer conversion index in an annual format. Send annual index to printer Prepare for new year.
A
No
A


C
Process master tape w/quarterly index program
RESULT
RESULT
Printout of quarterly index
Quarterly data set on mag. tape used in forming annual index
Check index for errors
Are there errors?
Yes
Use the Edit Master Program to correct the bad documents on master file
No
F
G

Flowchart of the System (cont'd.)

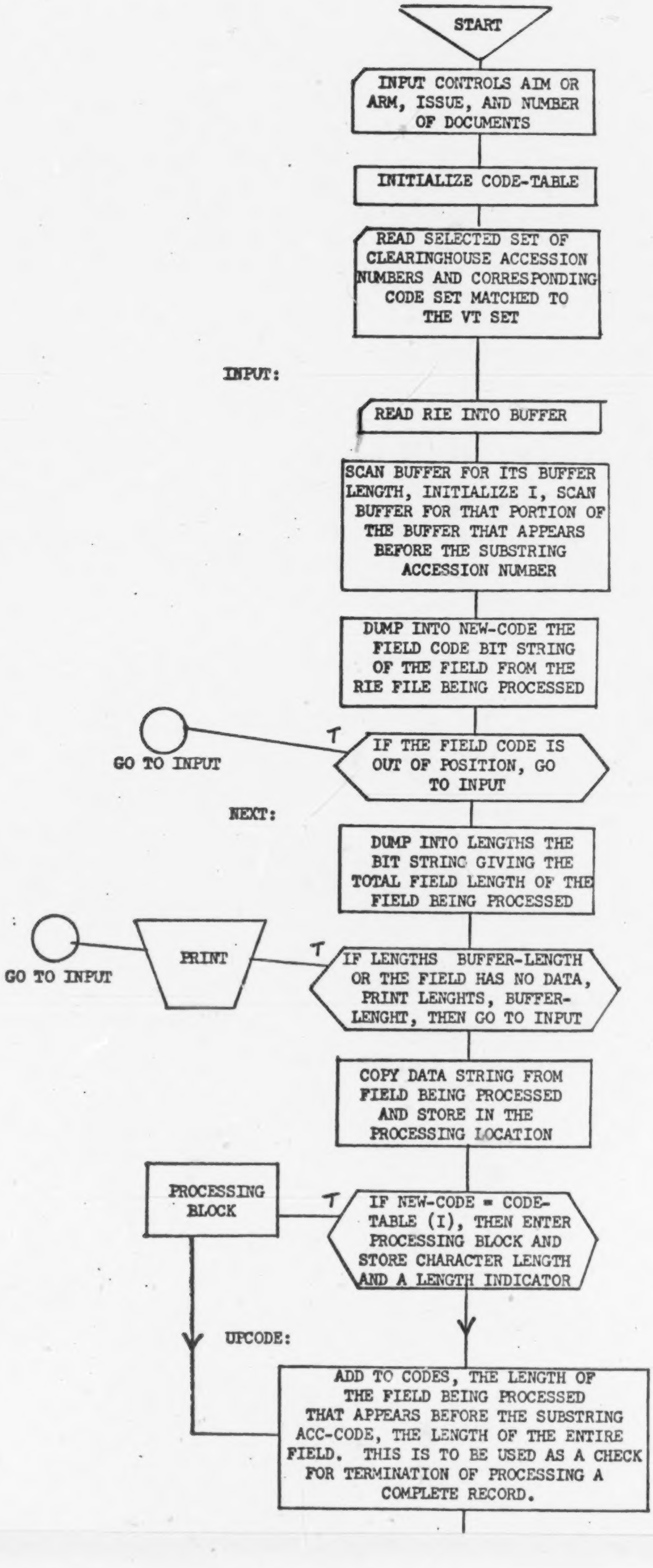

F
G
Send index printout to be published
RESULT
Edited master tape
Is it summer quarter?
No
A
C
Yes
Annual Procedure Followed for the Indexes in this Path
Fall Quarterly Data Set
Winter Quarterly Data Set
Spring Quarterly Data Set
Summer Quarterly Data Set
MERGING DATA SETS
The quarterly data sets on magnetic tape are merged to form one large set.
RESULT
Direct access to annual data set.
D
Process the data set w/the Annual Program belonging to this index.
RESULT
Annual Index Printout
H

Flowchart of the System (cont'd.)

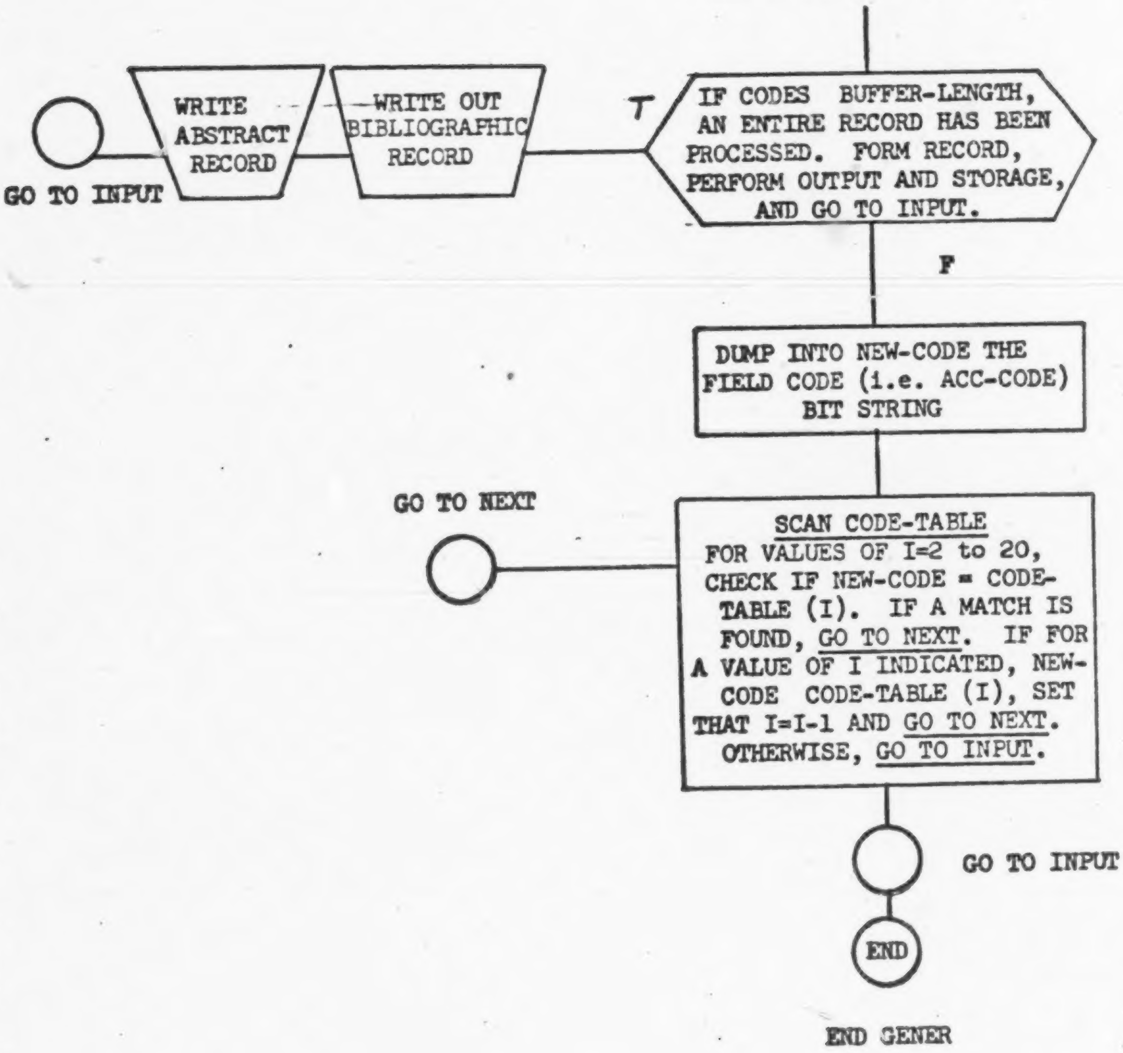

H
Are there errors in the printout?
Yes
Use the Edit Program for the specific index to edit the direct access data set.
RESULT
Updated annual direct access data set.
Update master tape using Master Edit so that master tape is always in accurate form
RESULT
Updated master tape
D
No
Send annual index for publication. Prepare to start a fresh year.
A

RIE TO AIM/ARM

RIE to AIM/ARM

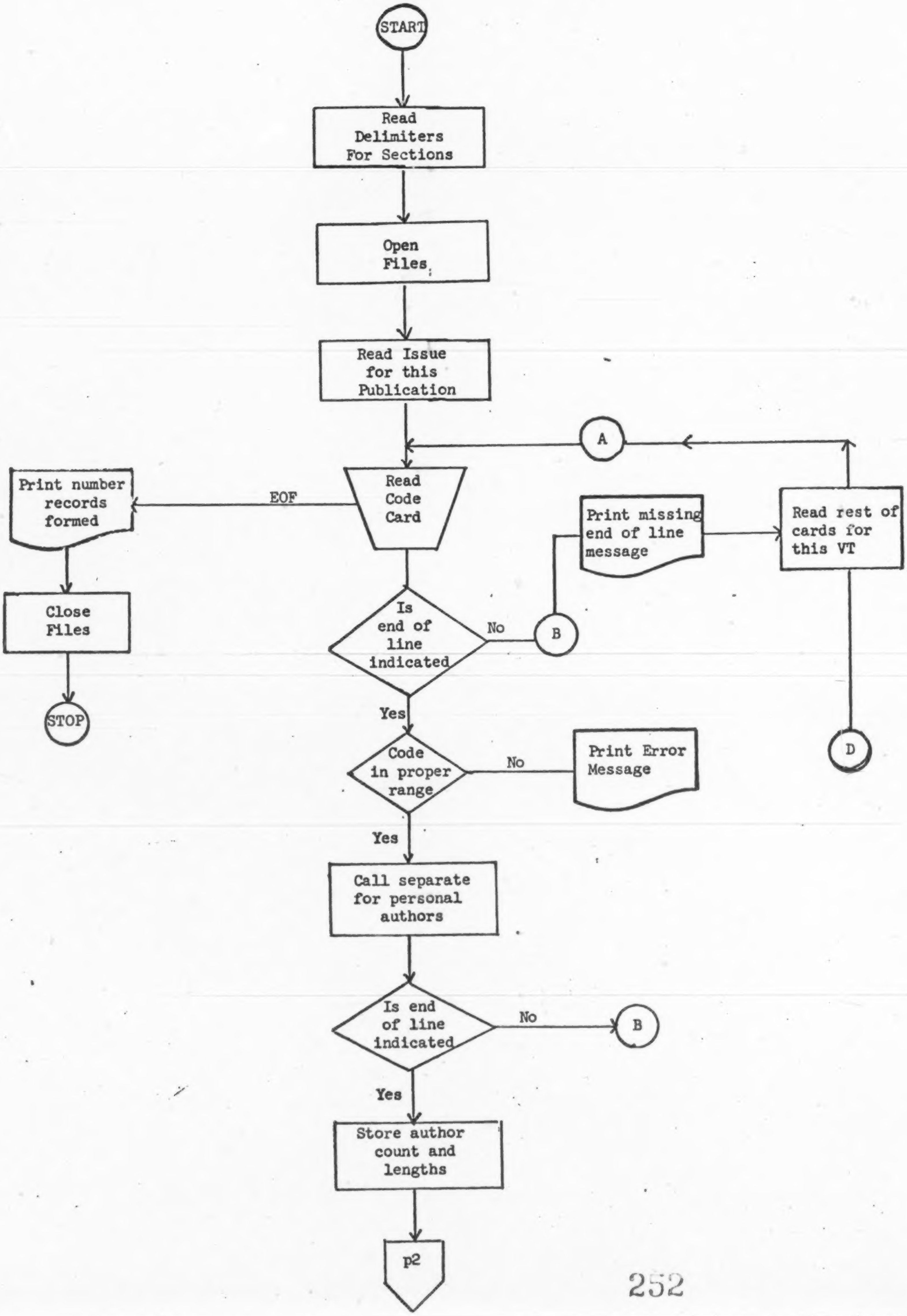

START
INPUT CONTROLS AIM OR ARM, ISSUE, AND NUMBER OF DOCUMENTS
INITIALIZE CODE-TABLE
READ SELECTED SET OF CLEARINGHOUSE ACCESSION NUMBERS AND CORRESPONDING CODE SET MATCHED TO THE VT SET
INPUT:
READ RIE INTO BUFFER
SCAN BUFFER FOR ITS BUFFER LENGTH, INITIALIZE I, SCAN BUFFER FOR THAT PORTION OF THE BUFFER THAT APPEARS BEFORE THE SUBSTRING ACCESSION NUMBER
DUMP INTO NEW-CODE THE FIELD CODE BIT STRING OF THE FIELD FROM THE RIE FILE BEING PROCESSED
GO TO INPUT
T
IF THE FIELD CODE IS OUT OF POSITION, GO TO INPUT
NEXT:
DUMP INTO LENGTHS THE BIT STRING GIVING THE TOTAL FIELD LENGTH OF THE FIELD BEING PROCESSED
GO TO INPUT
PRINT
T
IF LENGTHS BUFFER-LENGTH OR THE FIELD HAS NO DATA, PRINT LENGHTS, BUFFER-LENGHT, THEN GO TO INPUT
COPY DATA STRING FROM FIELD BEING PROCESSED AND STORE IN THE PROCESSING LOCATION
PROCESSING BLOCK
T
IF NEW-CODE = CODE-TABLE (I), THEN ENTER PROCESSING BLOCK AND STORE CHARACTER LENGTH AND A LENGTH INDICATOR
UPCODE:
ADD TO CODES, THE LENGTH OF THE FIELD BEING PROCESSED THAT APPEARS BEFORE THE SUBSTRING ACC-CODE, THE LENGTH OF THE ENTIRE FIELD. THIS IS TO BE USED AS A CHECK FOR TERMINATION OF PROCESSING A COMPLETE RECORD.

RIE to AIM/ARM (cont'd.)

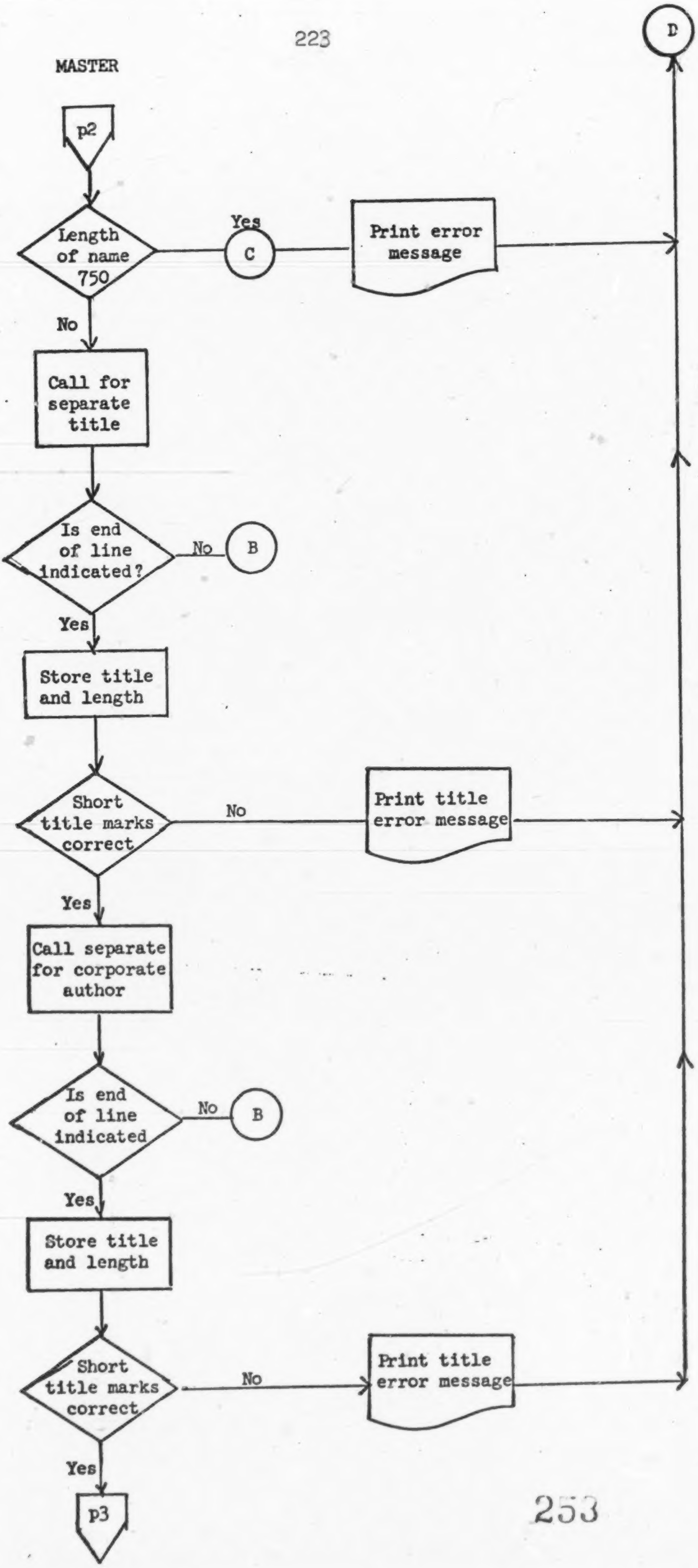

GO TO INPUT
WRITE ABSTRACT RECORD
WRITE OUT BIBLIOGRAPHIC RECORD
T
IF CODES BUFFER-LENGTH, AN ENTIRE RECORD HAS BEEN PROCESSED. FORM RECORD, PERFORM OUTPUT AND STORAGE, AND GO TO INPUT.
F
DUMP INTO NEW-CODE THE FIELD CODE (i.e. ACC-CODE) BIT STRING
GO TO NEXT
SCAN CODE-TABLE
FOR VALUES OF I=2 to 20, CHECK IF NEW-CODE = CODE-TABLE (I). IF A MATCH IS FOUND, GO TO NEXT. IF FOR A VALUE OF I INDICATED, NEW-CODE CODE-TABLE (I), SET THAT I=I-1 AND GO TO NEXT. OTHERWISE, GO TO INPUT.
GO TO INPUT
END
END GENER

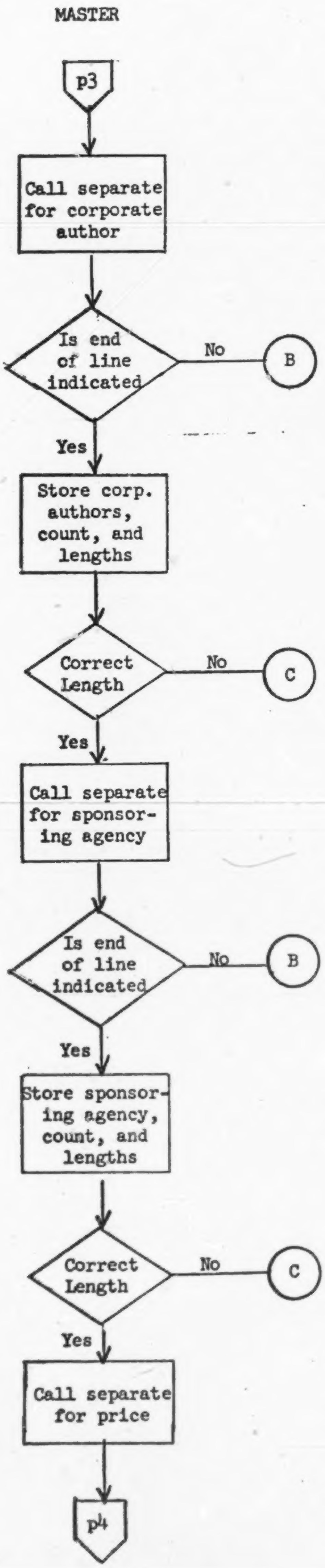
MASTER

MASTER

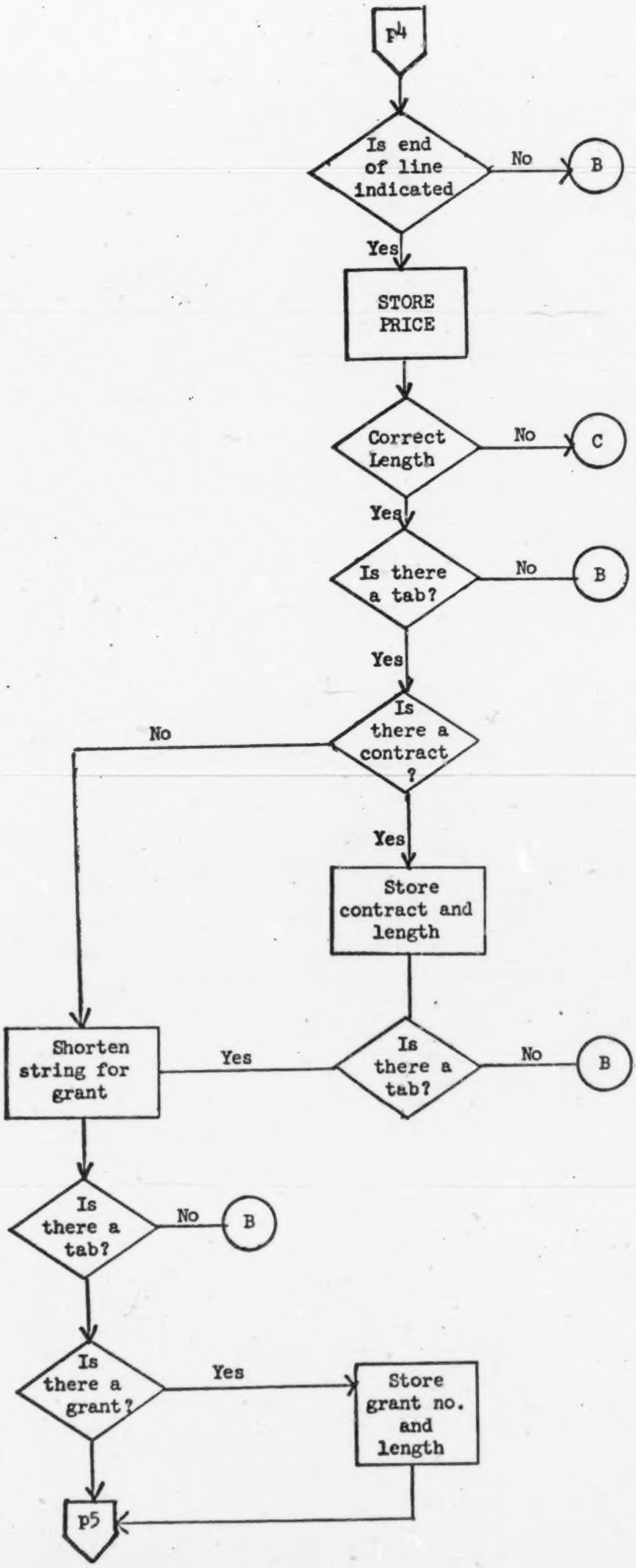

START
Read Delimiters For Sections
Open Files
Read Issue for this Publication
A
Read Code Card
EOF
Print number records formed
Close Files
STOP
Is end of line indicated
No
B
Print missing end of line message
Read rest of cards for this VT
D
Yes
Code in proper range
No
Print Error Message
Yes
Call separate for personal authors
Is end of line indicated
No
B
Yes
Store author count and lengths
p2

MASTER

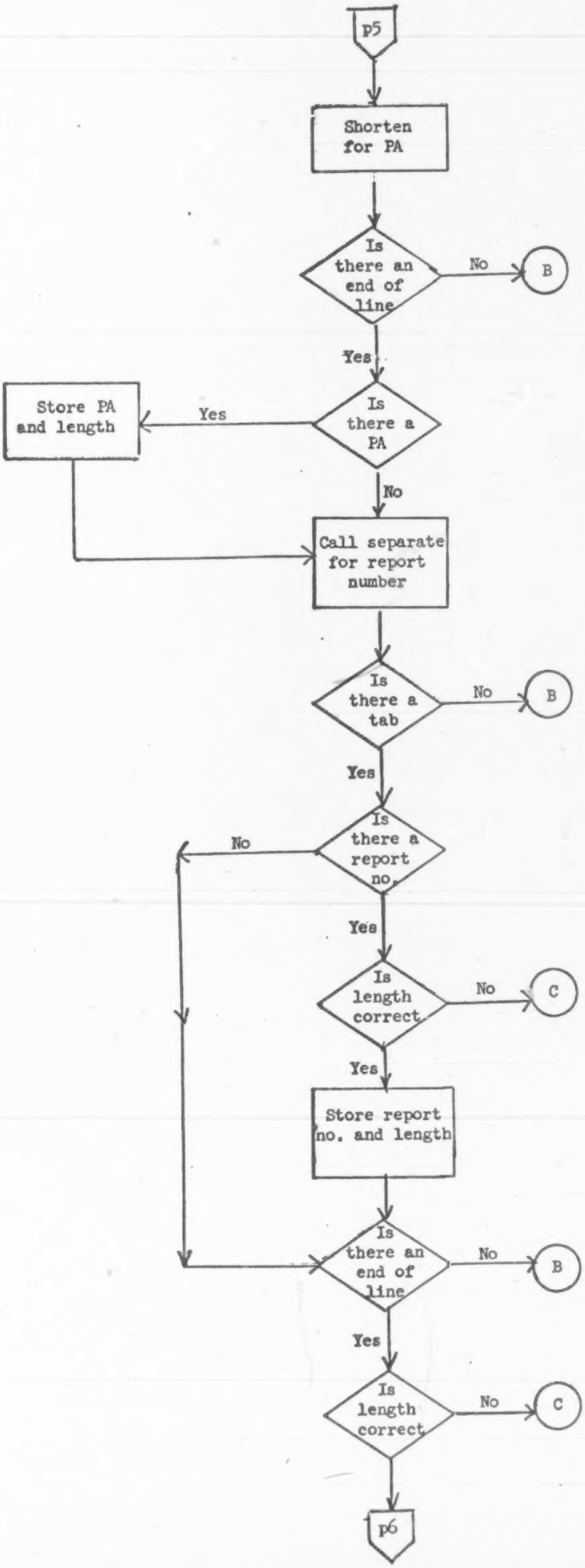

p2
Length of name 750
Yes
C
Print error message
D
No
Call for separate title
Is end of line indicated?
No
B
Yes
Store title and length
Short title marks correct
No
Print title error message
Yes
Call separate for corporate author
Is end of line indicated
No
B
Yes
Store title and length
Short title marks correct
No
Print title error message
Yes
p3


MASTER
p3
Call separate for corporate author
Is end of line indicated
No
B
Yes
Store corp. authors, count, and lengths
Correct Length
No
C
Yes
Call separate for sponsoring agency
Is end of line indicated
No
B
Yes
Store sponsoring agency, count, and lengths
Correct Length
No
C
Yes
Call separate for price
p4

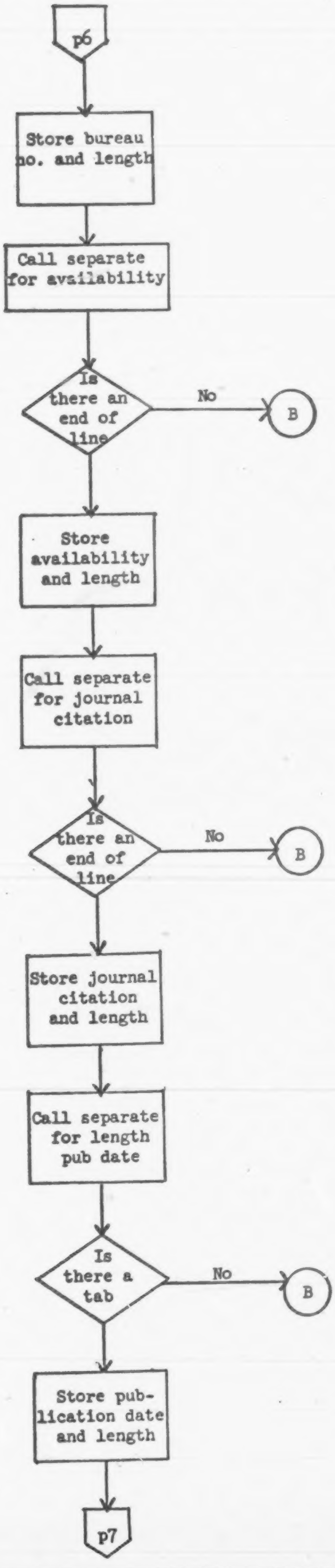
MASTER


P4
Is end of line indicated
No
B
Yes
STORE PRICE
Correct Length
No
C
Yes
Is there a tab?
No
B
Yes
Is there a contract ?
No
Yes
Store contract and length
Is there a tab?
No
B
Yes
Shorten string for grant
Is there a tab?
No
B
Is there a grant?
Yes
Store grant no. and length
p5

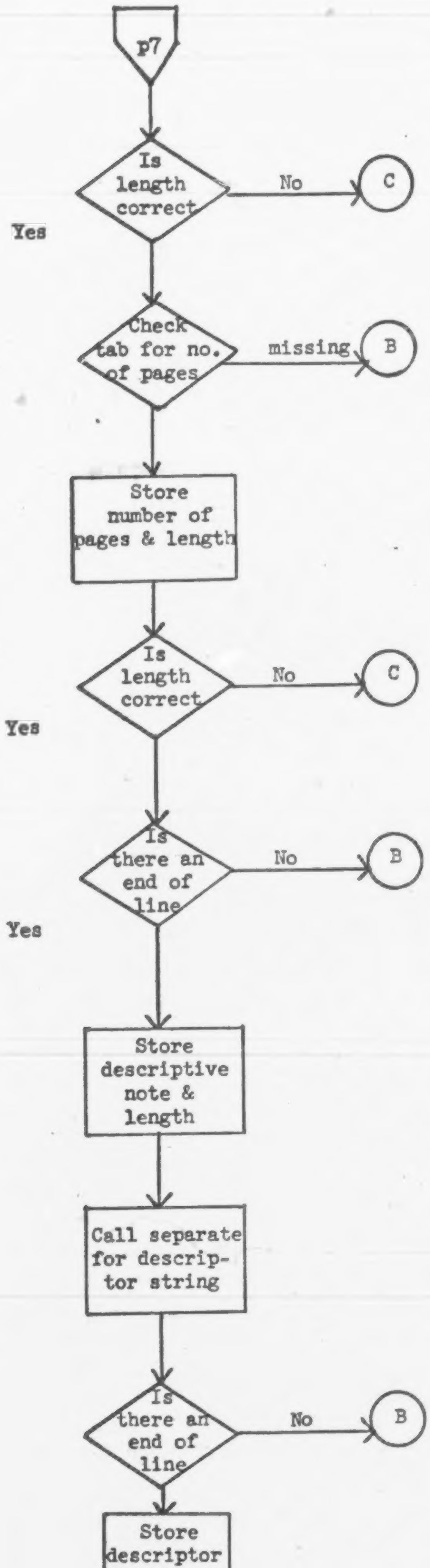
MASTER


p5
Shorten for PA
Is there an end of line
No
B
Yes
Is there a PA
Yes
Store PA and length
No
Call separate for report number
Is there a tab
No
B
Yes
Is there a report no.
No
Yes
Is length correct
No
C
Yes
Store report no. and length
Is there an end of line
No
B
Yes
Is length correct
No
C
p6

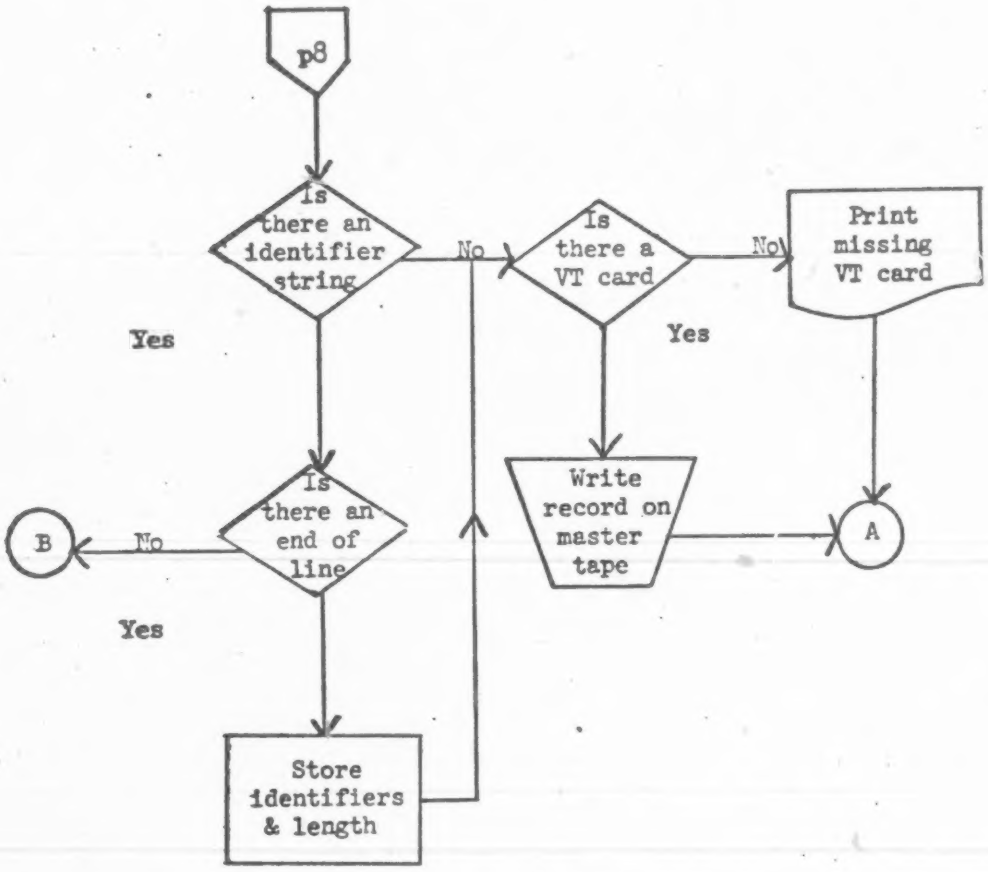
MASTER


p6
Store bureau no. and length
Call separate for availability
Is there an end of line
No
B
Store availability and length
Call separate for journal citation
Is there an end of line
No
B
Store journal citation and length
Call separate for length pub date
Is there a tab
No
B
Store publication date and length
p7

MASTER


p7
Is length correct
No
C
Yes
Check tab for no. of pages
missing
B
Store number of pages & length
Is length correct
No
C
Yes
Is there an end of line
No
B
Yes
Store descriptive note & length
Call separate for descriptor string
Is there an end of line
No
B
Store descriptor string & length
p8

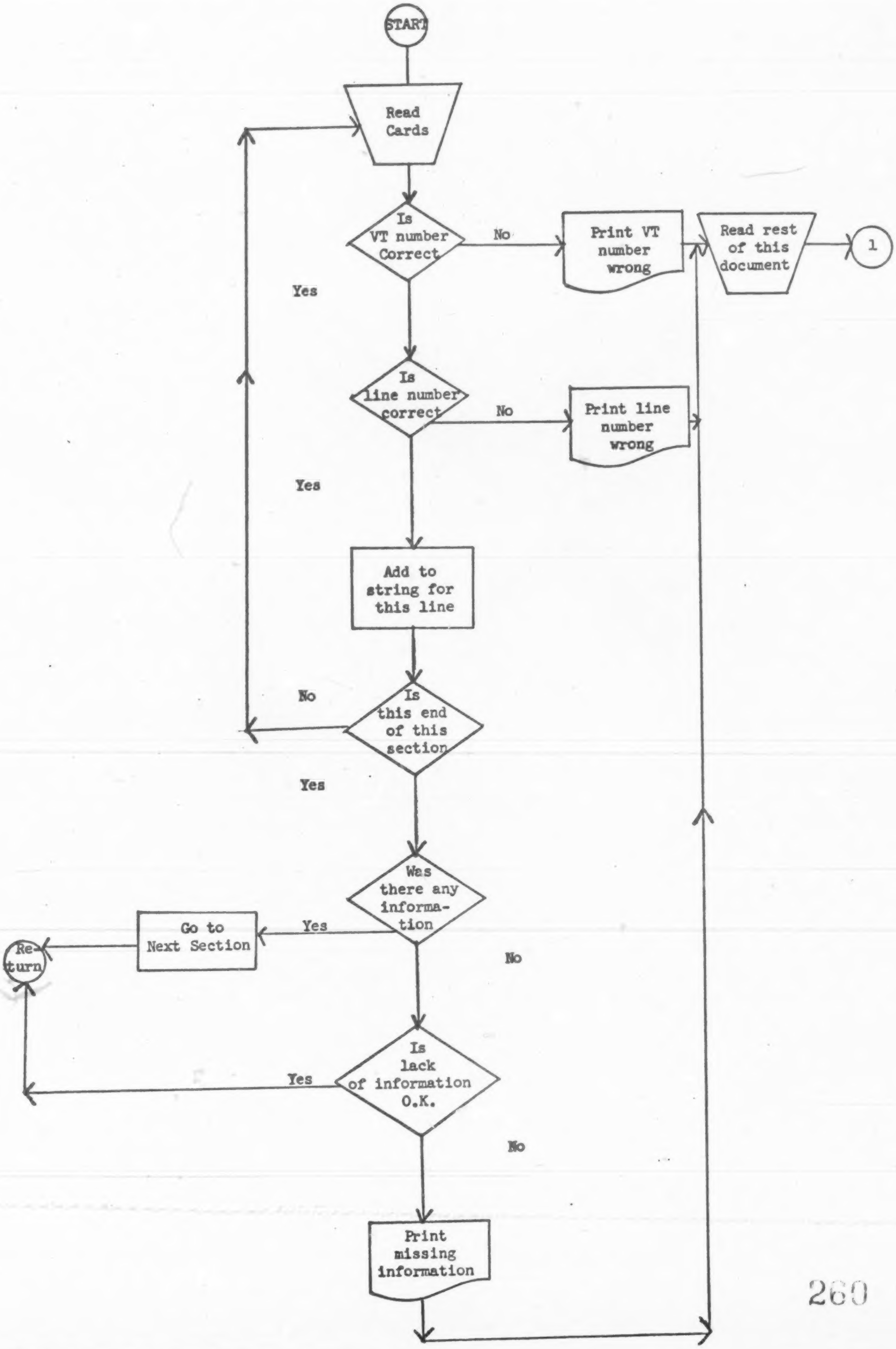
MASTER


p8
Is there an identifier string
Yes
No
Is there a VT card
No
Print missing VT card
Yes
Write record on master tape
Is there an end of line
No
B
Yes
Store identifiers & length
A

MASTER

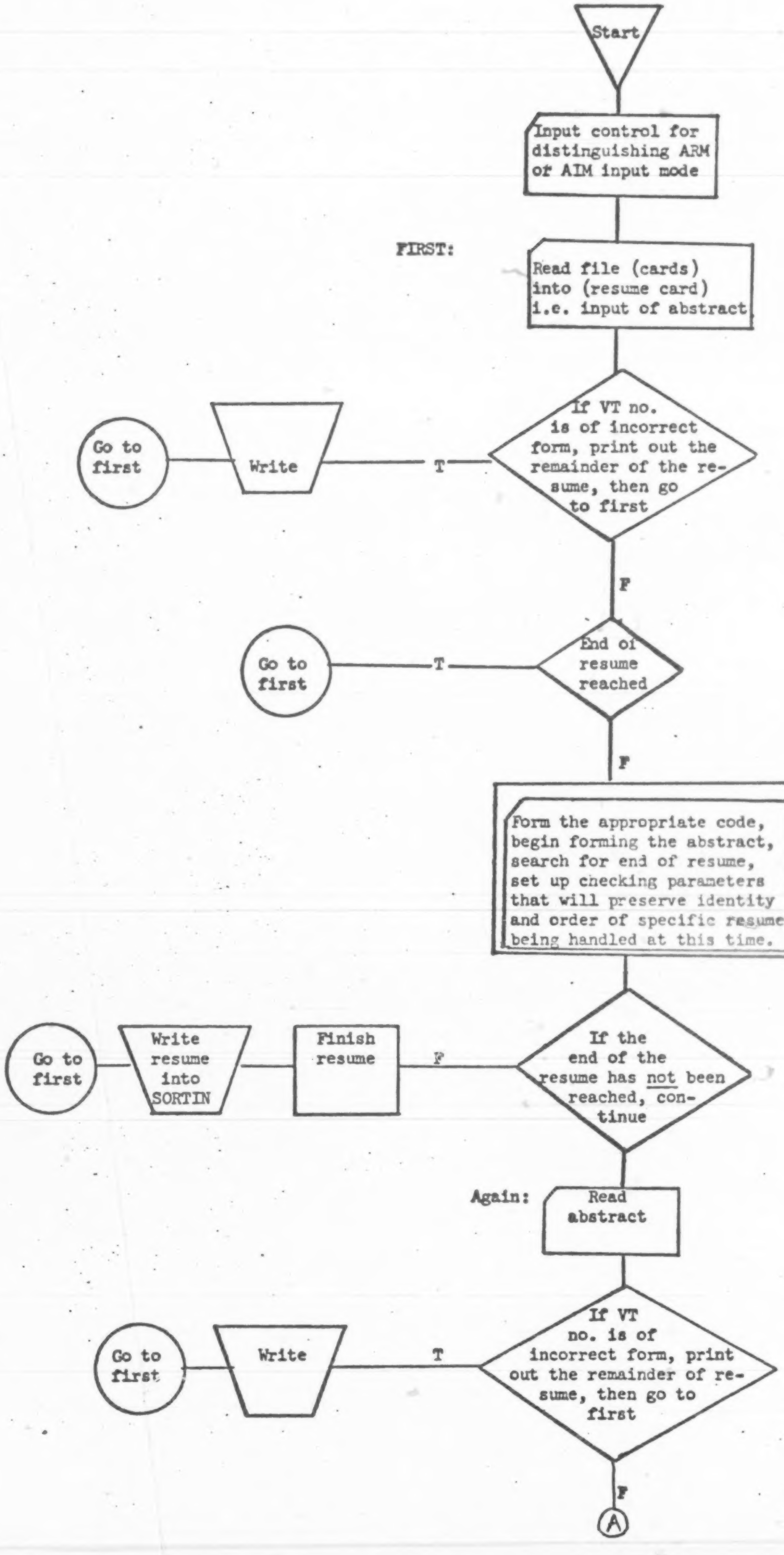
230

SEPARATE

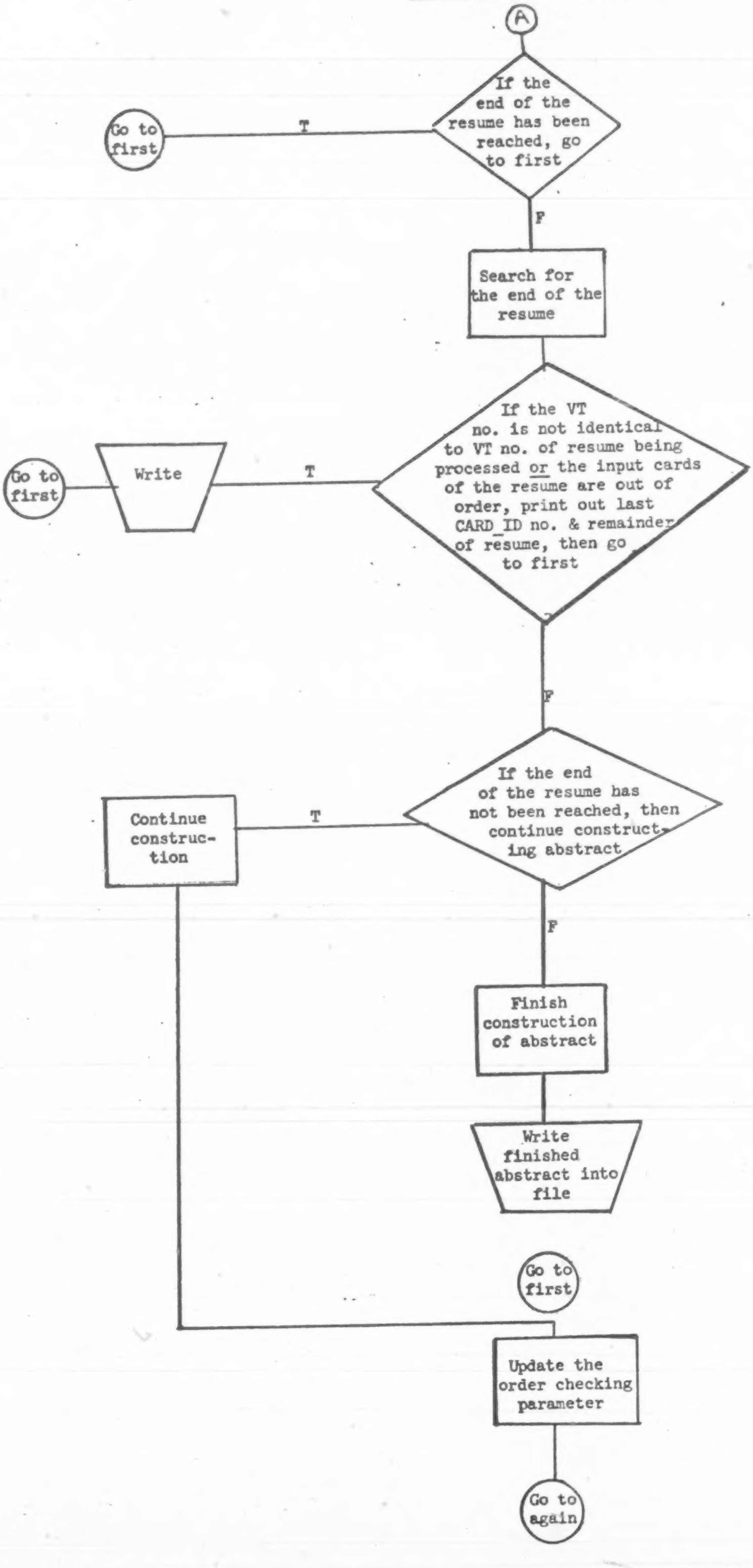

START
Read Cards
Is VT number Correct
No
Print VT number wrong
Read rest of this document
1
Yes
Is line number correct
No
Print line number wrong
Yes
Add to string for this line
No
Is this end of this section
Yes
Was there any information
Yes
Go to Next Section
Return
No
Is lack of information O.K.
Yes
No
Print missing information

ABSTRACT

Create Abstract Records

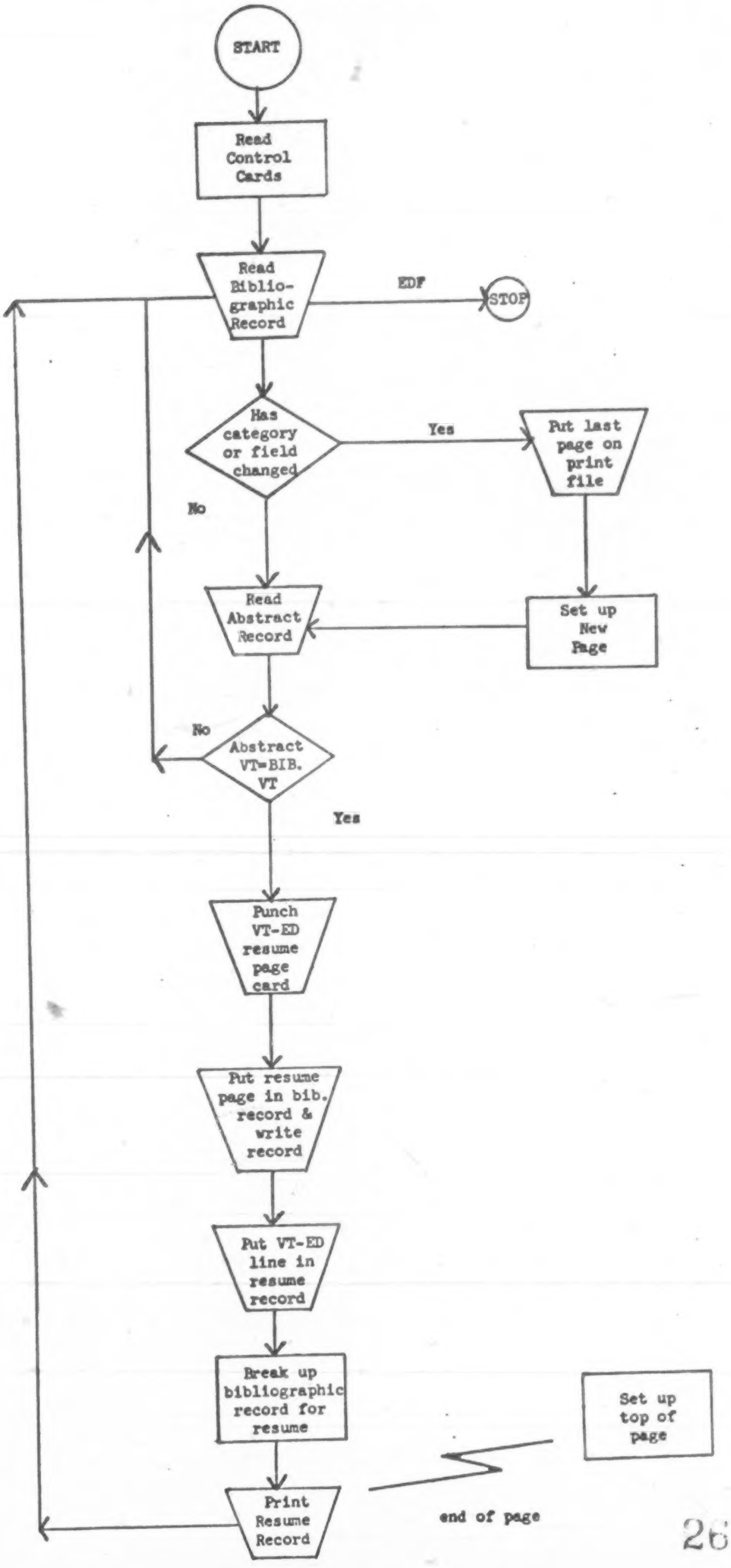

Start
Input control for distinguishing ARM or AIM input mode
FIRST:
Read file (cards) into (resume card) i.e. input of abstract
If VT no. is of incorrect form, print out the remainder of the resume, then go to first
T
Write
Go to first
F
End of resume reached
T
Go to first
F
Form the appropriate code, begin forming the abstract, search for end of resume, set up checking parameters that will preserve identity and order of specific resume being handled at this time.
If the end of the resume has not been reached, continue
F
Finish resume
Write resume into SORTIN
Go to first
Again:
Read abstract
If VT no. is of incorrect form, print out the remainder of resume, then go to first
T
Write
Go to first
F
A

Create Abstract Records (cont'd.)


A
If the end of the resume has been reached, go to first
T
Go to first
F
Search for the end of the resume
If the VT no. is not identical to VT no. of resume being processed or the input cards of the resume are out of order, print out last CARD_ID no. & remainder of resume, then go to first
T
Write
Go to first
F
If the end of the resume has not been reached, then continue constructing abstract
T
Continue construction
F
Finish construction of abstract
Write finished abstract into file
Go to first
Update the order checking parameter
Go to again

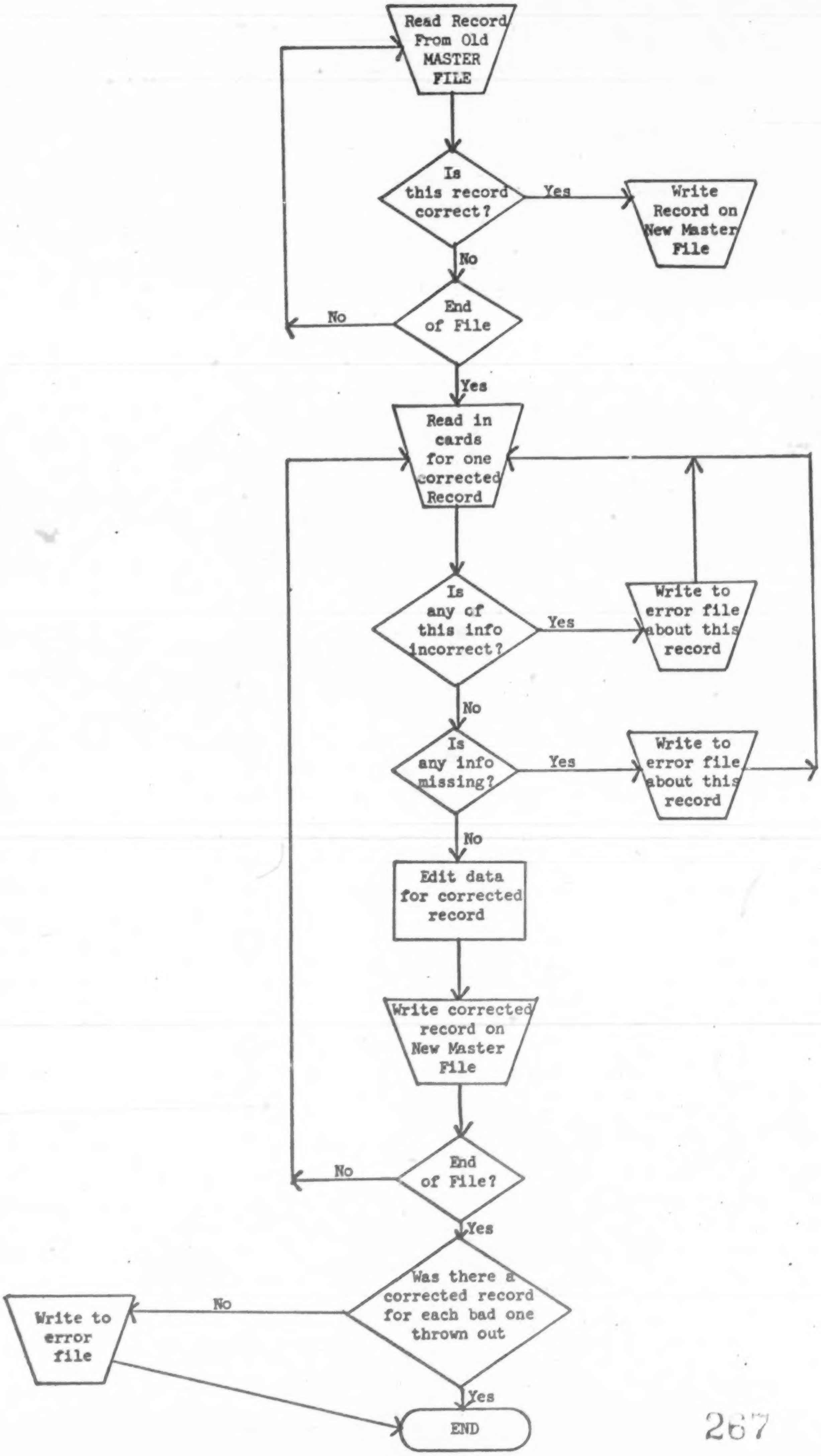
RESUME PRINT

RESUME PRINT


START
Read Control Cards
Read Biblio-graphic Record
EOF
STOP
Has category or field changed
Yes
No
Put last page on print file
Set up New Page
Read Abstract Record
No
Abstract VT=BIB. VT
Yes
Punch VT-ED resume page card
Put resume page in bib. record & write record
Put VT-ED line in resume record
Break up bibliographic record for resume
Set up top of page
Print Resume Record
end of page

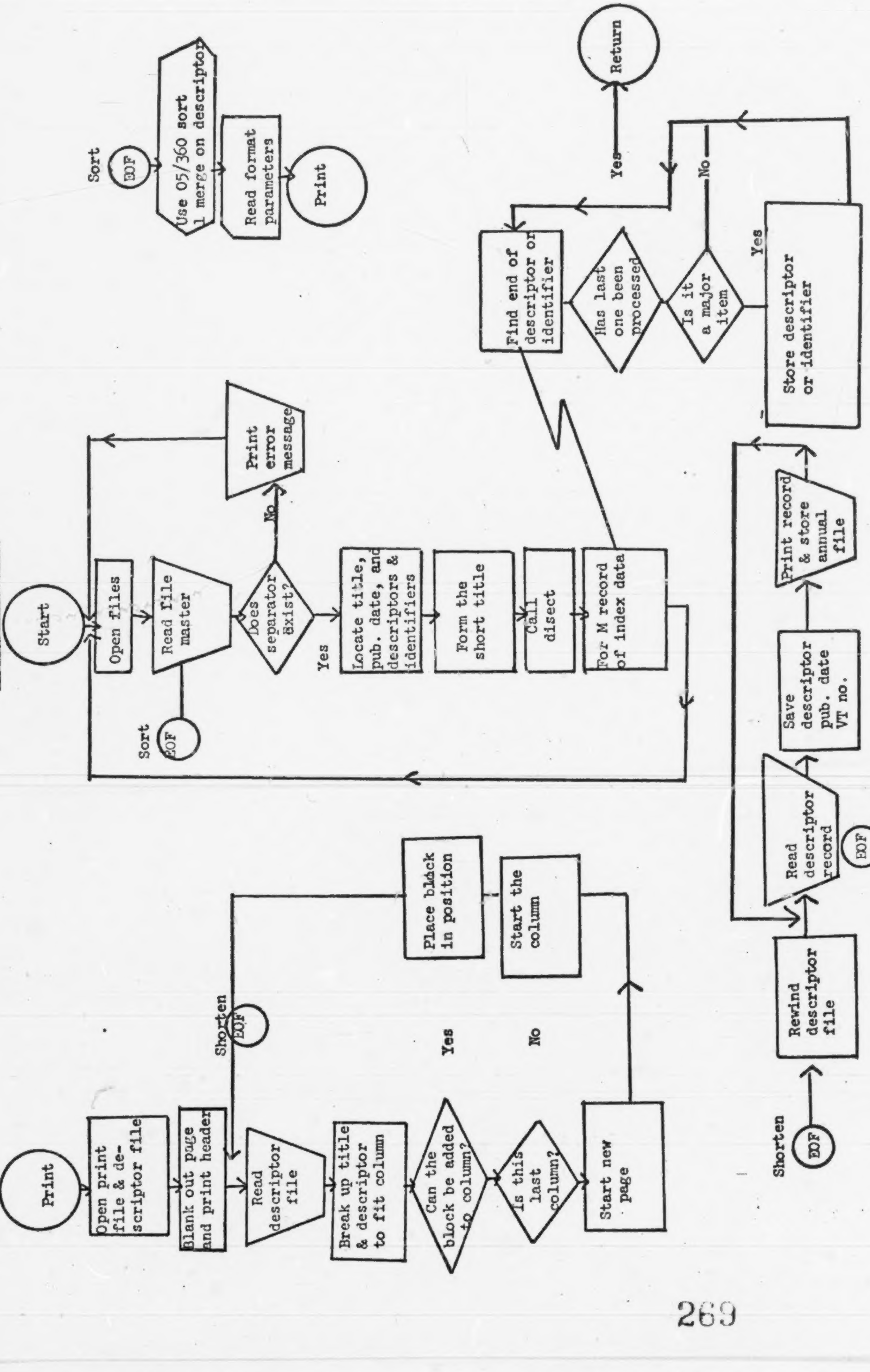
MASTER EDIT

MASTER EDIT

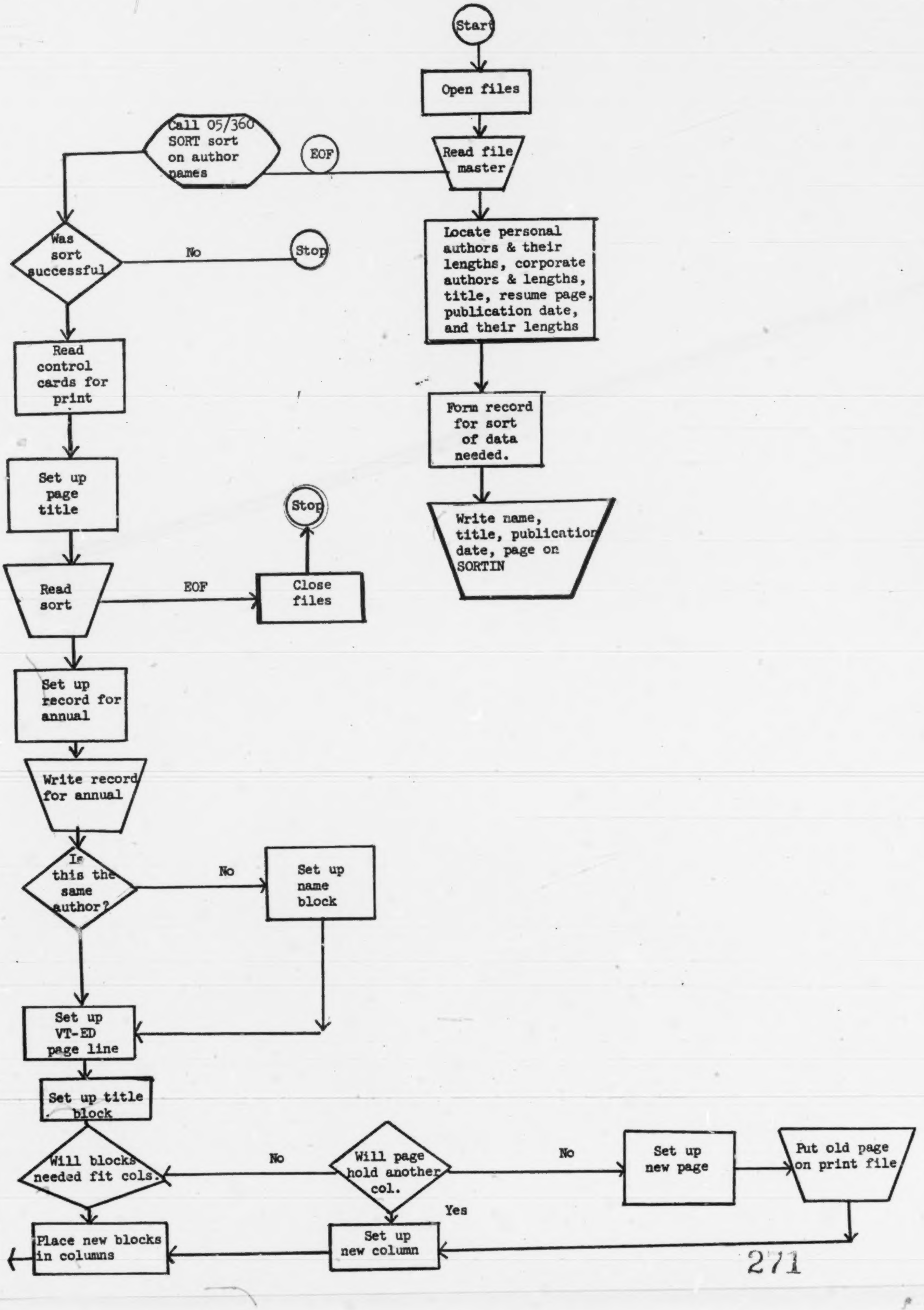

Read Record From Old MASTER FILE
Is this record correct?
Yes
Write Record on New Master File
No
End of File
No
Yes
Read in cards for one corrected Record
Is any of this info incorrect?
Yes
Write to error file about this record
No
Is any info missing?
Yes
Write to error file about this record
No
Edit data for corrected record
Write corrected record on New Master File
End of File?
No
Yes
Was there a corrected record for each bad one thrown out
No
Write to error file
Yes
END

SUBJECT QUARTERLY

QUARTERLY SUBJECT

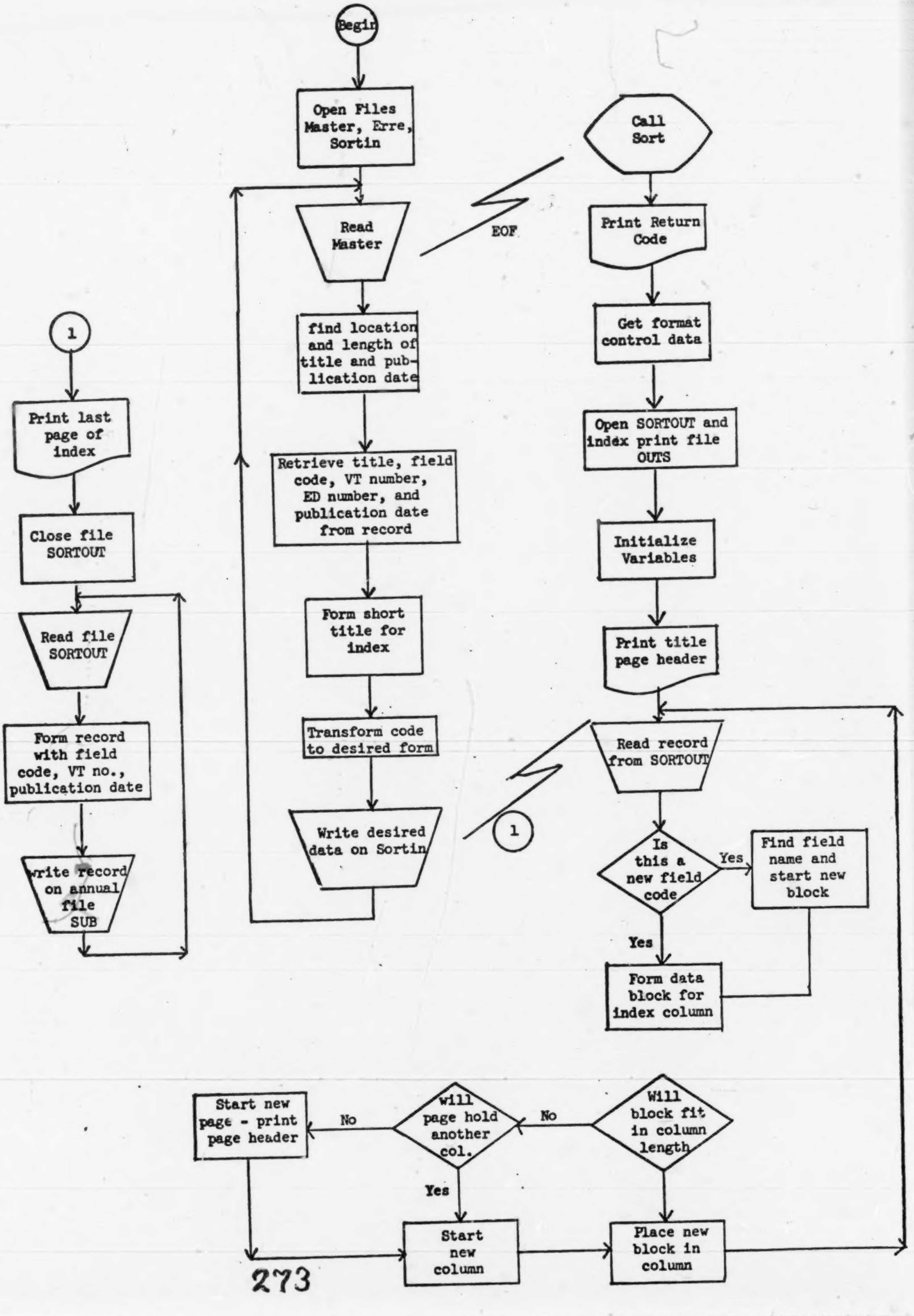

Start
Open files
Read file master
Sort EOF
Does separator exist?
No
Print error message
Yes
Locate title, pub. date, and descriptors & identifiers
Form the short title
Call disect
For M record of index data
Find end of descriptor or identifier
Has last one been processed
Yes
Return
Is it a major item
No
Yes
Store descriptor or identifier
Sort EOF
Use OS/360 sort 1 merge on descriptor
Read format parameters
Print
Print
Open print file & descriptor file
Blank out page and print header
Read descriptor file
Shorten EOF
Break up title & descriptor to fit column
Can the block be added to column?
Yes
Place block in position
Is this last column?
No
Start the column
Start new page
Shorten EOF
Rewind descriptor file
Read descriptor record
Save descriptor pub. date VT no.
Print record & store annual file
EOF
Close Annual File
Stop

AUTHOR QUARTERLY


Start
Open files
Read file master
EOF
Call 05/360 SORT sort on author names
Locate personal authors & their lengths, corporate authors & lengths, title, resume page, publication date, and their lengths
Form record for sort of data needed.
Write name, title, publication date, page on SORTIN
Was sort successful
No
Stop
Read control cards for print
Set up page title
Read sort
EOF
Close files
Stop
Set up record for annual
Write record for annual
Is this the same author?
No
Set up name block
Set up VT-ED page line
Set up title block
Will blocks needed fit cols.
No
Will page hold another col.
No
Set up new page
Put old page on print file
Yes
Set up new column
Place new blocks in columns

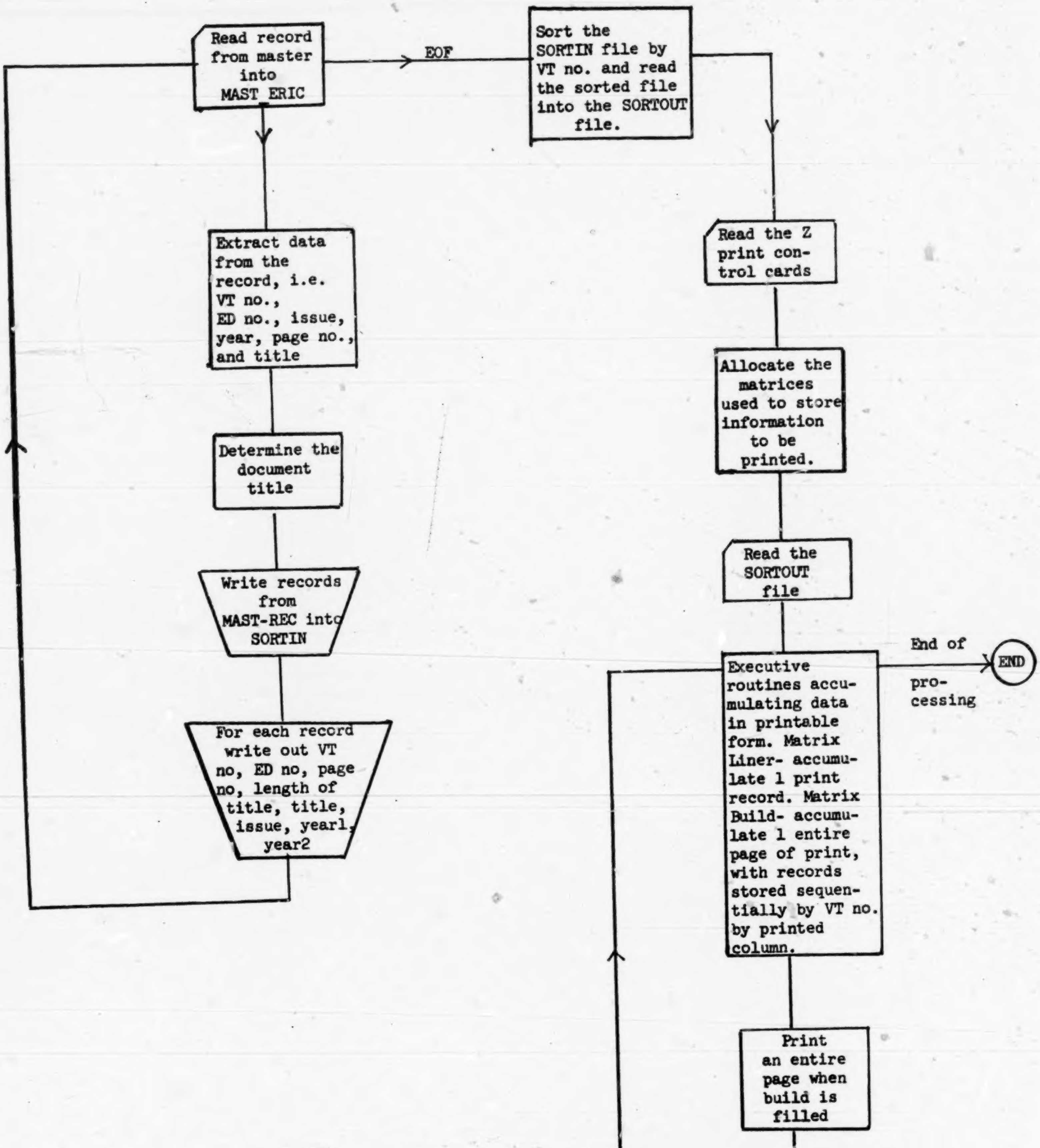
SUPPORTING SERVICES QUARTERLY

Supporting Services Quarterly


Begin
Open Files Master, Erre, Sortin
Read Master
EOF
find location and length of title and publication date
Retrieve title, field code, VT number, ED number, and publication date from record
Form short title for index
Transform code to desired form
Write desired data on Sortin
1
Call Sort
Print Return Code
Get format control data
Open SORTOUT and index print file OUTS
Initialize Variables
Print title page header
Read record from SORTOUT
Is this a new field code
Yes
Find field name and start new block
Yes
Form data block for index column
Will block fit in column length
No
will page hold another col.
No
Start new page - print page header
Yes
Start new column
Place new block in column
1
Print last page of index
Close file SORTOUT
Read file SORTOUT
Form record with field code, VT no., publication date
write record on annual file SUB

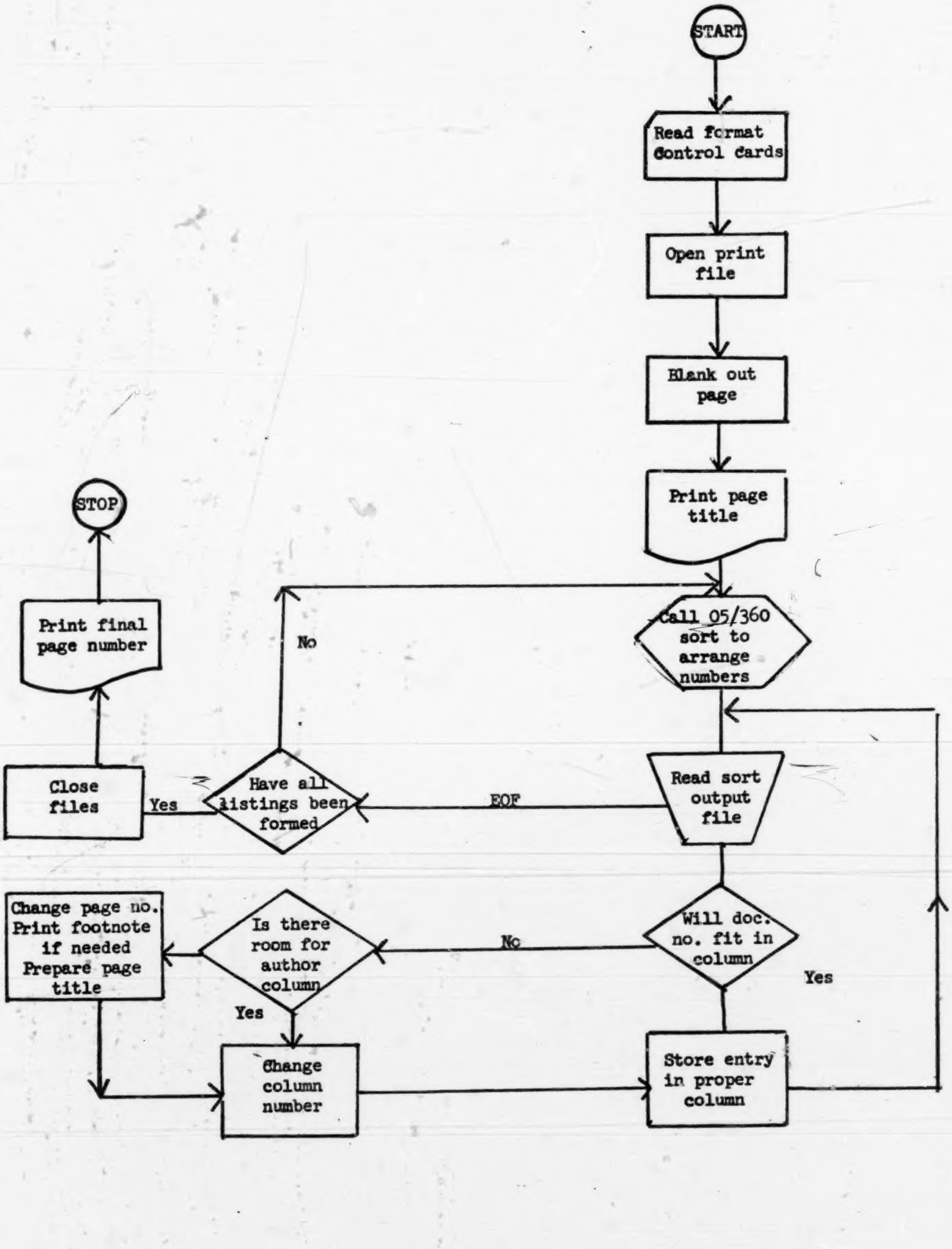
DOCUMENT NUMBER QUARTERLY

Document Number Quarterly


Read record from master into MAST ERIC
EOF
Sort the SORTIN file by VT no. and read the sorted file into the SORTOUT file.
Extract data from the record, i.e. VT no., ED no., issue, year, page no., and title
Determine the document title
Write records from MAST-REC into SORTIN
For each record write out VT no, ED no, page no, length of title, title, issue, year1, year2
Read the Z print control cards
Allocate the matrices used to store information to be printed.
Read the SORTOUT file
Executive routines accumulating data in printable form. Matrix Liner- accumulate 1 print record. Matrix Build- accumulate 1 entire page of print, with records stored sequentially by VT no. by printed column.
End of processing
END
Print an entire page when build is filled
Reinitialize matrix build

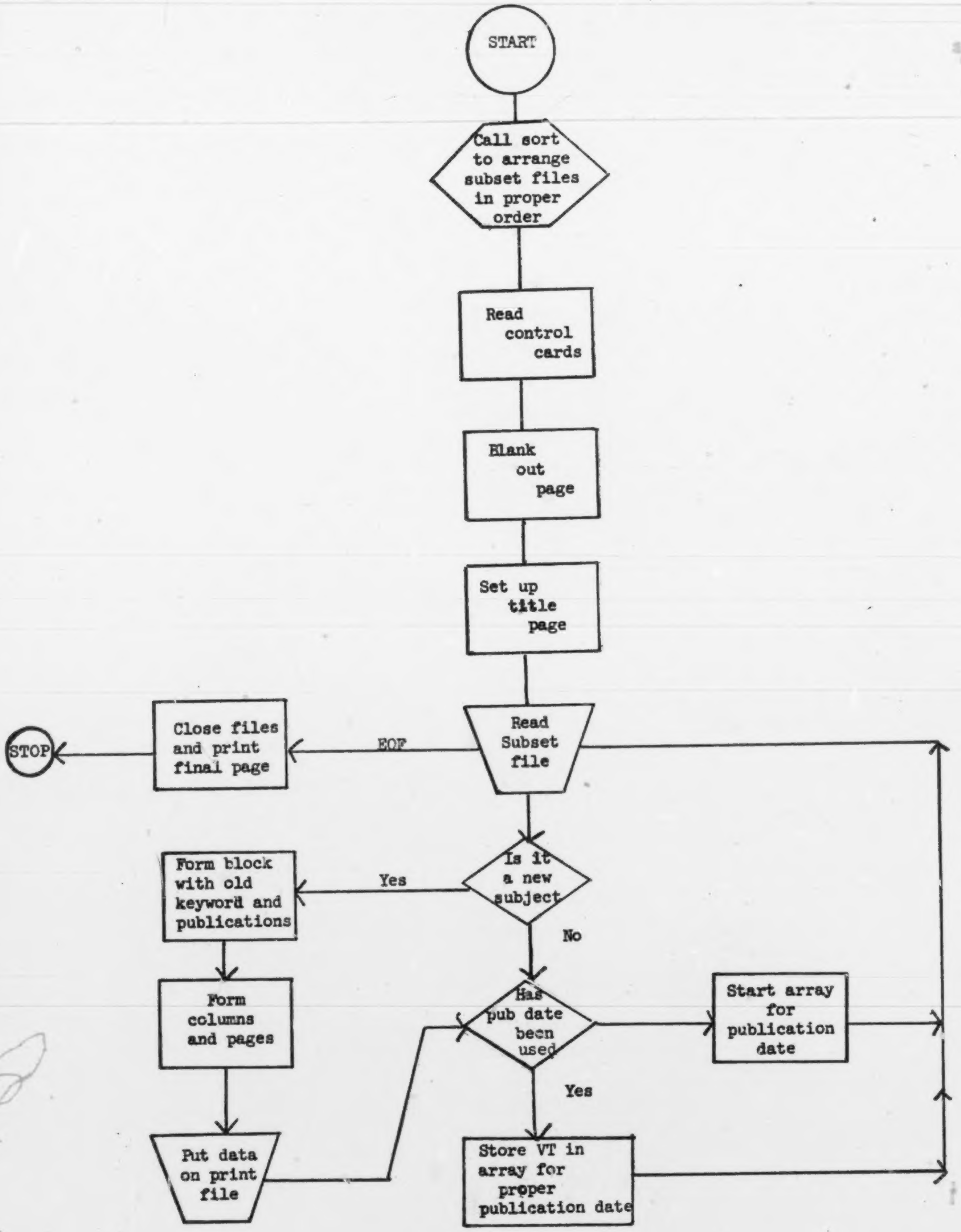
CONVERSION OF DOCUMENT NUMBER

CONVERSION OF DOCUMENT NUMBER

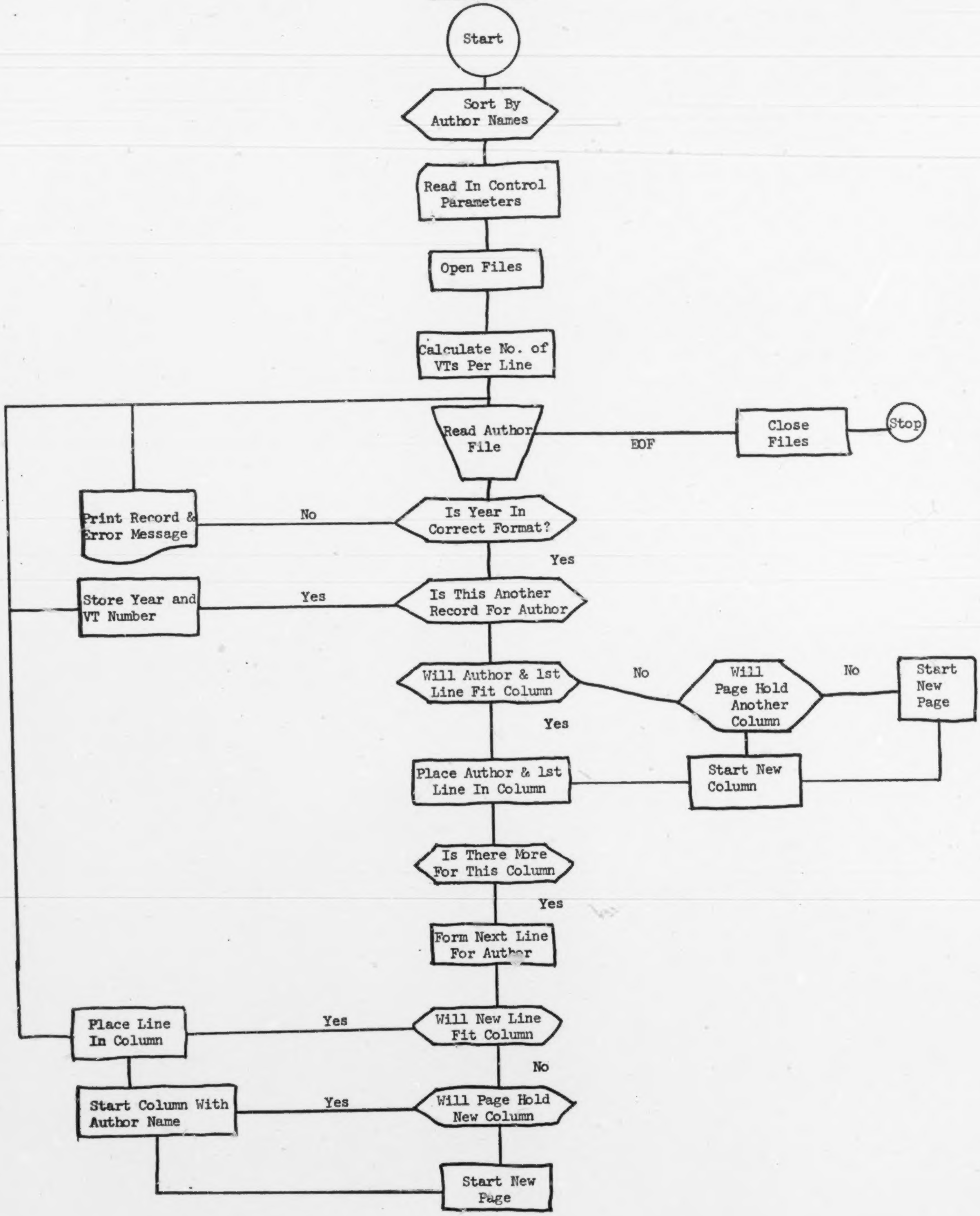

START
Read format control cards
Open print file
Blank out page
Print page title
Call OS/360 sort to arrange numbers
Read sort output file
EOF
Have all listings been formed
No
Yes
Close files
Print final page number
STOP
Will doc. no. fit in column
No
Yes
Is there room for author column
Yes
Change page no. Print footnote if needed Prepare page title
Change column number
Store entry in proper column

SUBJECT ANNUAL

SUBJECT ANNUAL


START
Call sort to arrange subset files in proper order
Read control cards
Blank out page
Set up title page
Read Subset file
EOF
Close files and print final page
STOP
Is it a new subject
Yes
No
Form block with old keyword and publications
Form columns and pages
Put data on print file
Has pub date been used
Start array for publication date
Yes
Store VT in array for proper publication date

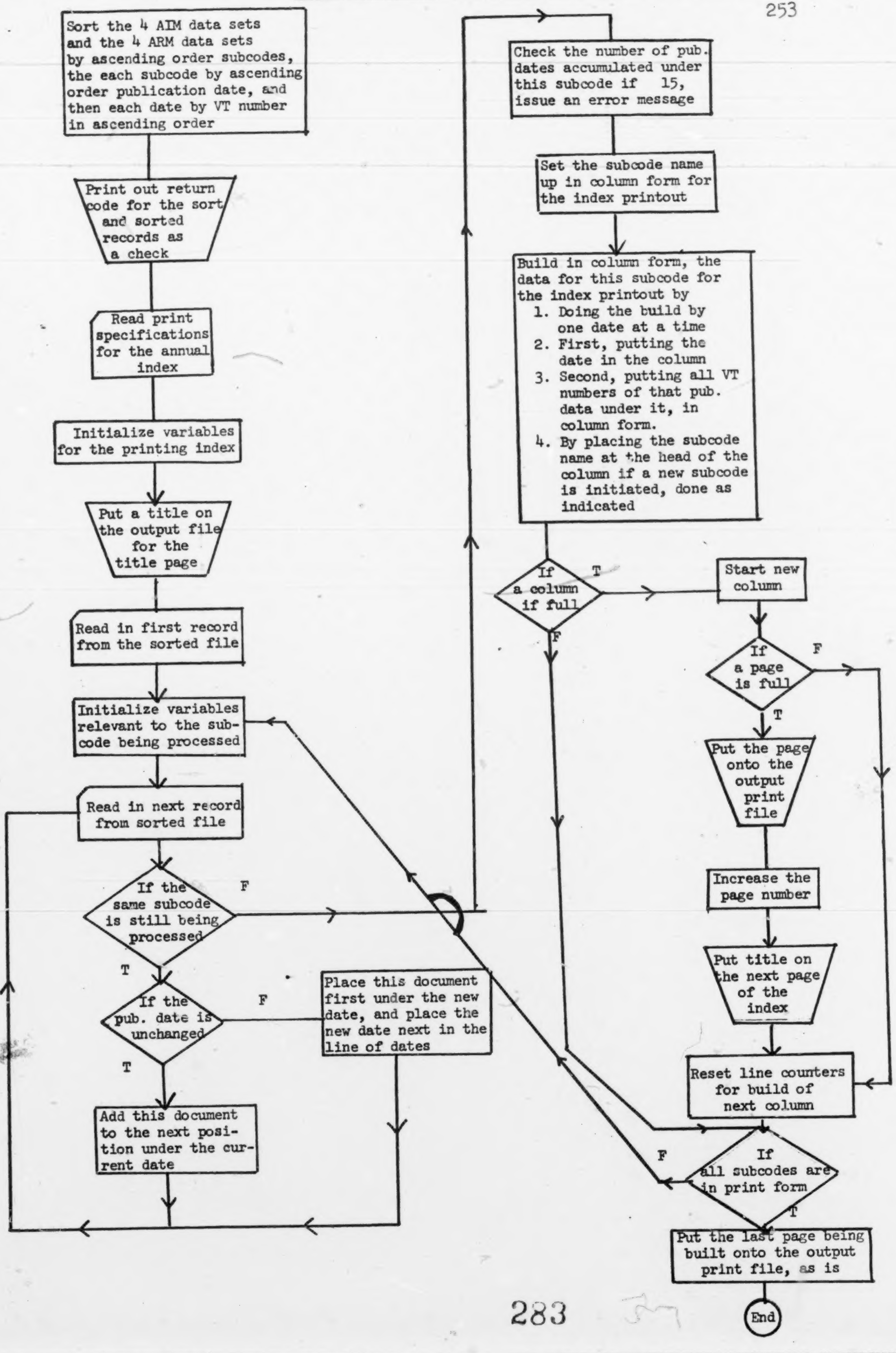
AUTHOR ANNUAL


Start
Sort By Author Names
Read In Control Parameters
Open Files
Calculate No. of VTs Per Line
Read Author File
EOF
Close Files
Stop
Print Record & Error Message
No
Is Year In Correct Format?
Yes
Store Year and VT Number
Yes
Is This Another Record For Author
Will Author & 1st Line Fit Column
No
Will Page Hold Another Column
No
Start New Page
Yes
Place Author & 1st Line In Column
Start New Column
Is There More For This Column
Yes
Form Next Line For Author
Place Line In Column
Yes
Will New Line Fit Column
No
Start Column With Author Name
Yes
Will Page Hold New Column
Start New Page

AUTHOR ANNUAL

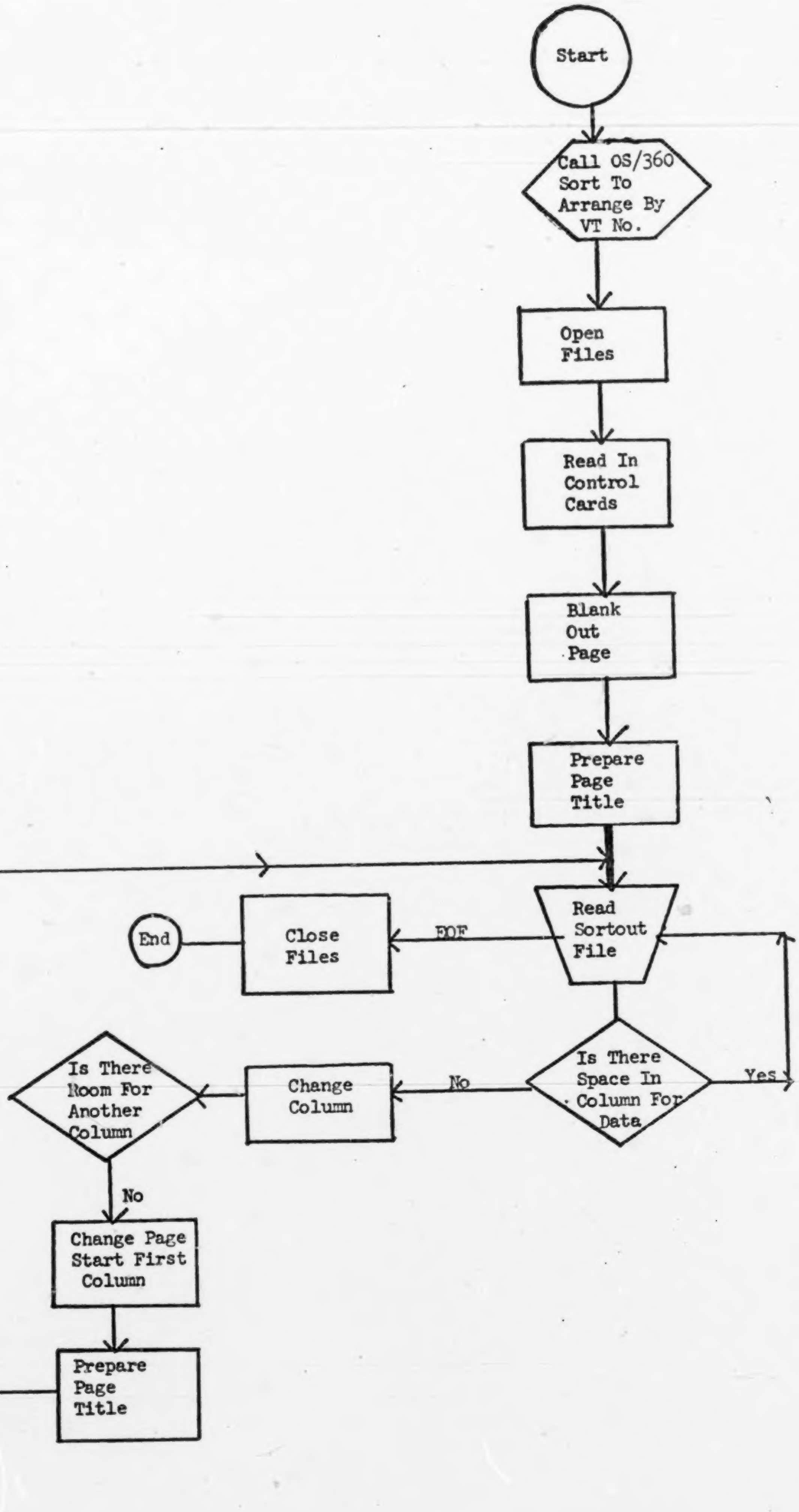
SUPPORTING SERVICES ANNUAL

FLOW CHART FOR ANNUAL SUPPORTING SERVICES INDEX PROGRAM


Sort the 4 AIM data sets and the 4 ARM data sets by ascending order subcodes, the each subcode by ascending order publication date, and then each date by VT number in ascending order
Print out return code for the sort and sorted records as a check
Read print specifications for the annual index
Initialize variables for the printing index
Put a title on the output file for the title page
Read in first record from the sorted file
Initialize variables relevant to the subcode being processed
Read in next record from sorted file
If the same subcode is still being processed
F
T
If the pub. date is unchanged
F
T
Place this document first under the new date, and place the new date next in the line of dates
Add this document to the next position under the current date
Check the number of pub. dates accumulated under this subcode if 15, issue an error message
Set the subcode name up in column form for the index printout
Build in column form, the data for this subcode for the index printout by
1. Doing the build by one date at a time
2. First, putting the date in the column
3. Second, putting all VT numbers of that pub. data under it, in column form.
4. By placing the subcode name at the head of the column if a new subcode is initiated, done as indicated
If a column if full
T
F
Start new column
If a page is full
F
T
Put the page onto the output print file
Increase the page number
Put title on the next page of the index
Reset line counters for build of next column
If all subcodes are in print form
F
T
Put the last page being built onto the output print file, as is
End

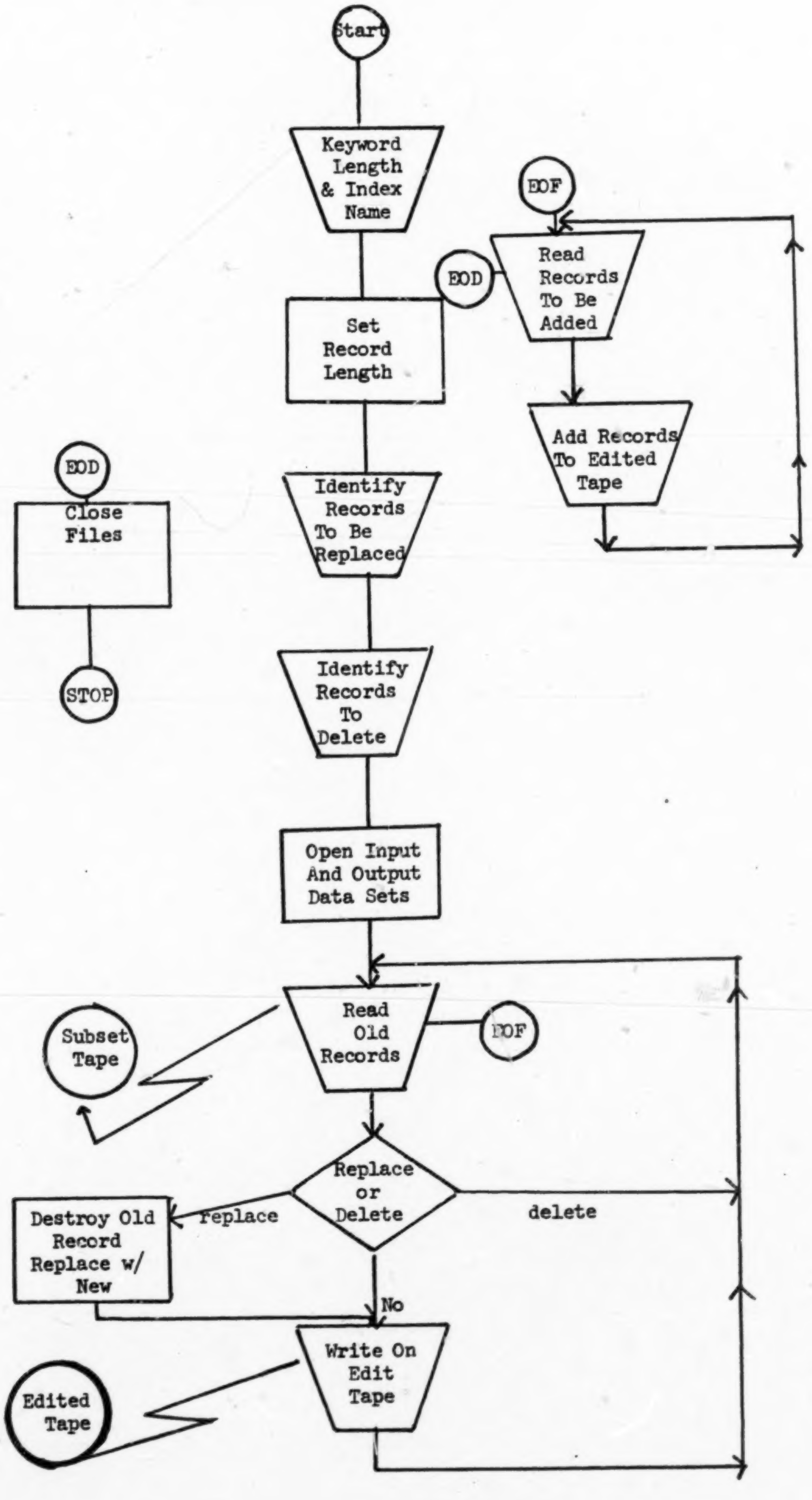
DOCUMENT NUMBER ANNUAL

DOCUMENT NUMBER ANNUAL


Start
Call OS/360 Sort To Arrange By VT No.
Open Files
Read In Control Cards
Blank Out Page
Prepare Page Title
Read Sortout File
EOF
Close Files
End
Is There Space In Column For Data
Yes
No
Change Column
Is There Room For Another Column
No
Change Page Start First Column
Prepare Page Title

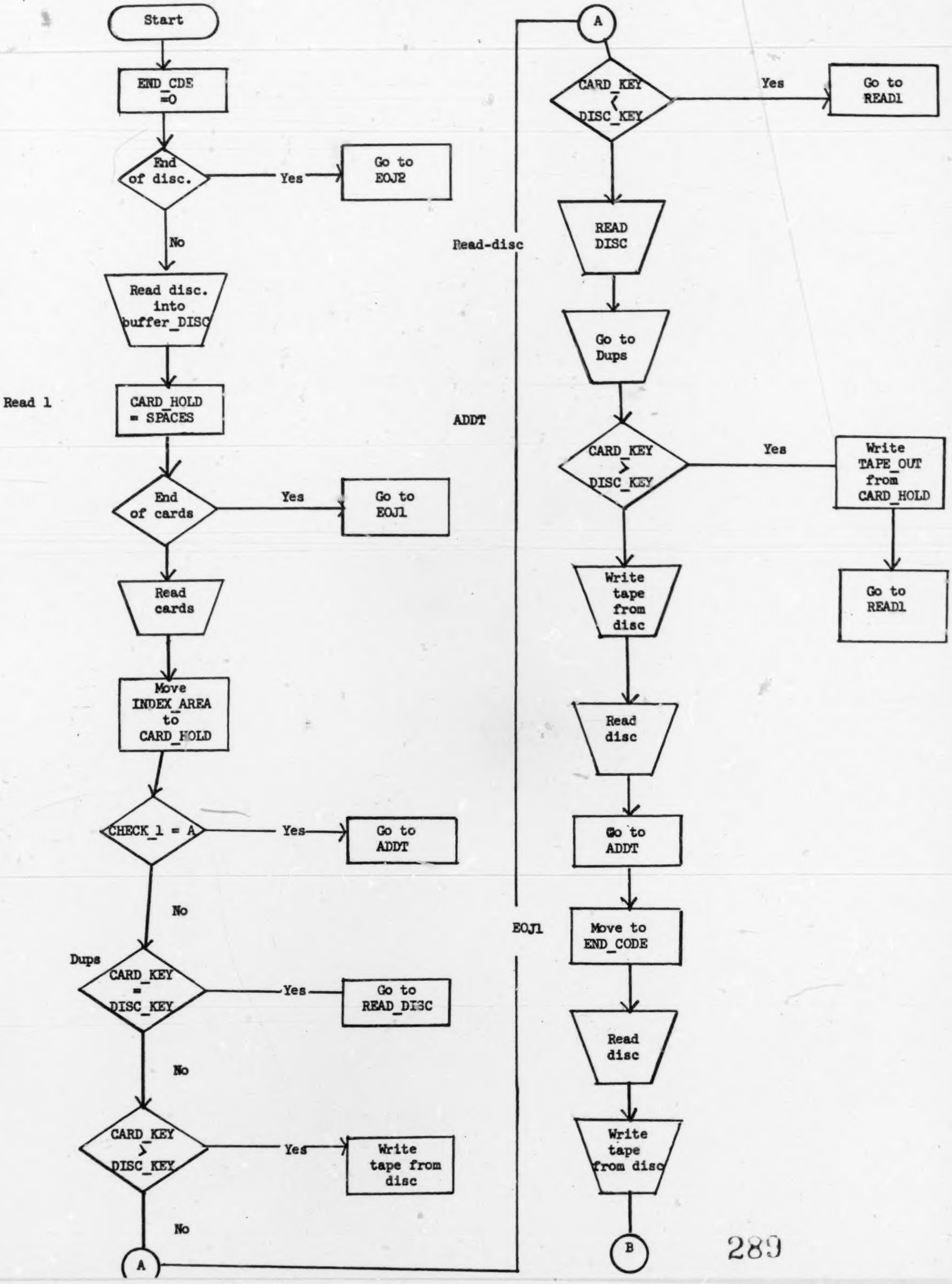
EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICES

EDIT OF SUBJECT, AUTHOR, AND SUPPORTING SERVICE

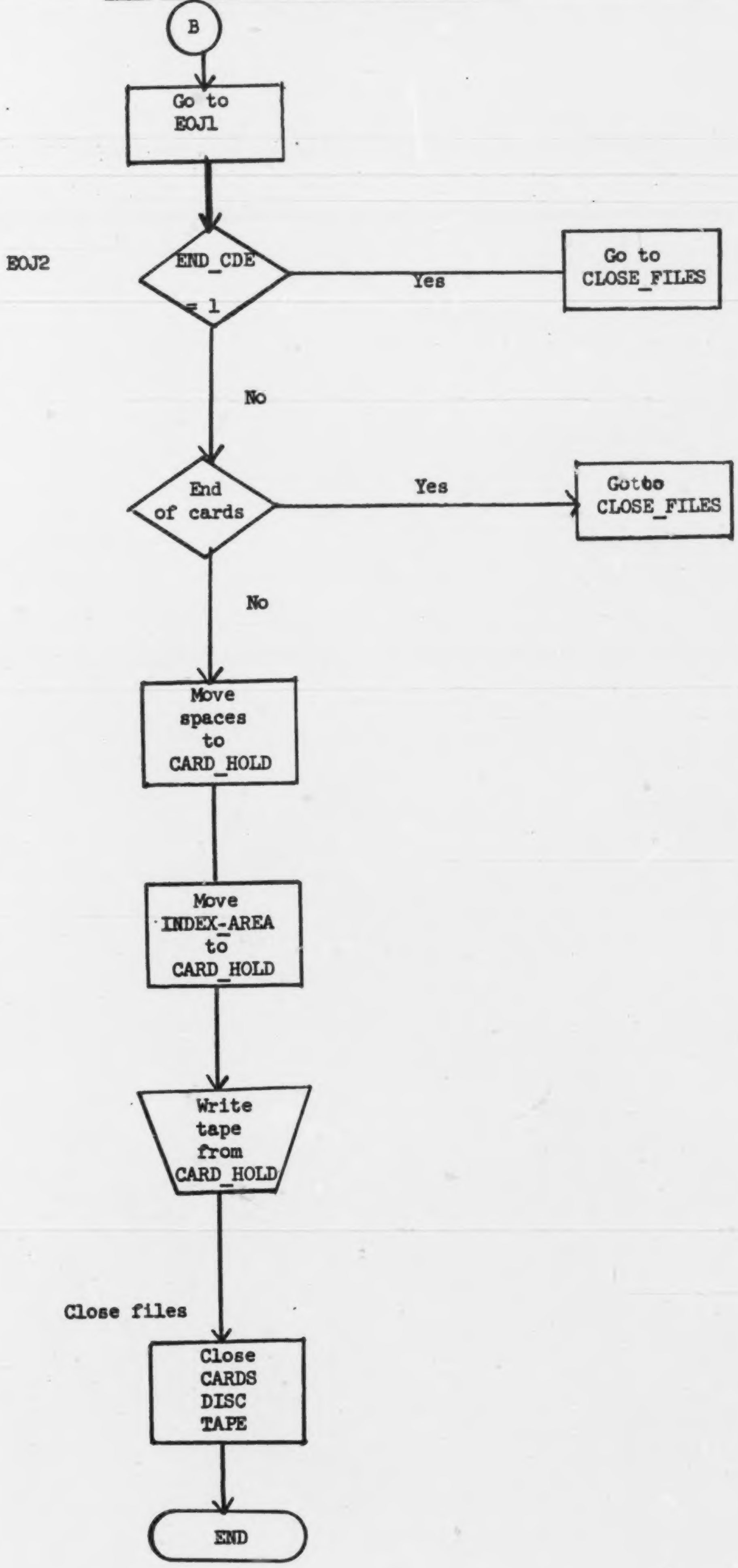

Start
Keyword Length & Index Name
Set Record Length
Identify Records To Be Replaced
Identify Records To Delete
Open Input And Output Data Sets
Read Old Records
Subset Tape
EOF
Replace or Delete
replace
delete
Destroy Old Record Replace w/ New
No
Write On Edit Tape
Edited Tape
EOF
EOD
Read Records To Be Added
Add Records To Edited Tape
EOD
Close Files
STOP

DOCUMENT NUMBER EDIT

Edit for Document Number

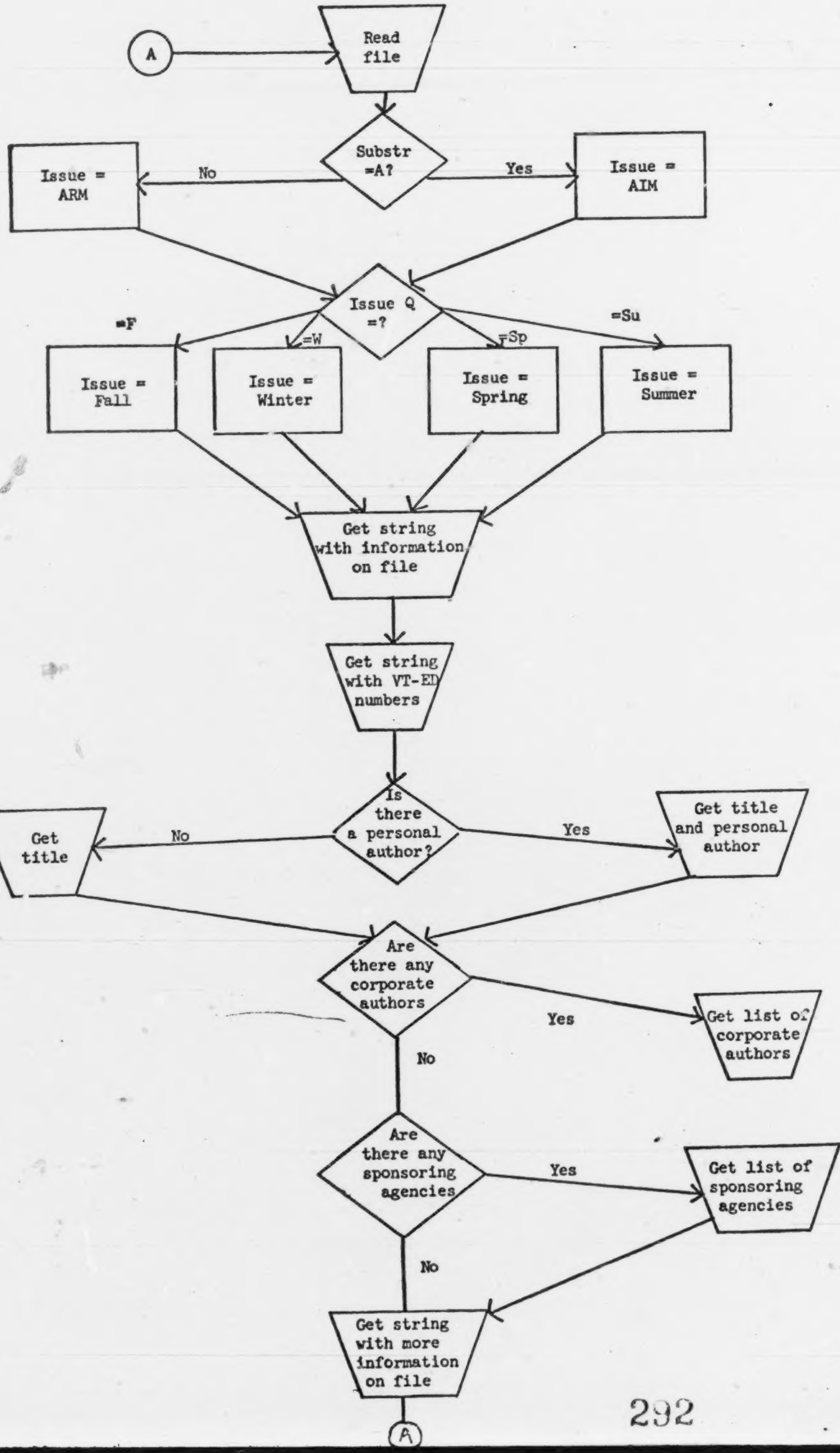

Start
END_CDE =0
End of disc.
Yes
Go to EOJ2
No
Read disc. into buffer_DISC
Read 1
CARD_HOLD = SPACES
End of cards
Yes
Go to EOJ1
Read cards
Move INDEX_AREA to CARD_HOLD
CHECK_1 = A
Yes
Go to ADDT
No
Dups
CARD_KEY = DISC_KEY
Yes
Go to READ_DISC
No
CARD_KEY > DISC_KEY
Yes
Write tape from disc
No
A
Read-disc
ADDT
EOJ1
A
CARD_KEY < DISC_KEY
Yes
Go to READ1
READ DISC
Go to Dups
CARD_KEY > DISC_KEY
Yes
Write TAPE_OUT from CARD_HOLD
Go to READ1
Write tape from disc
Read disc
Go to ADDT
Move to END_CODE
Read disc
Write tape from disc
B

Edit for Document Number (cont'd.)

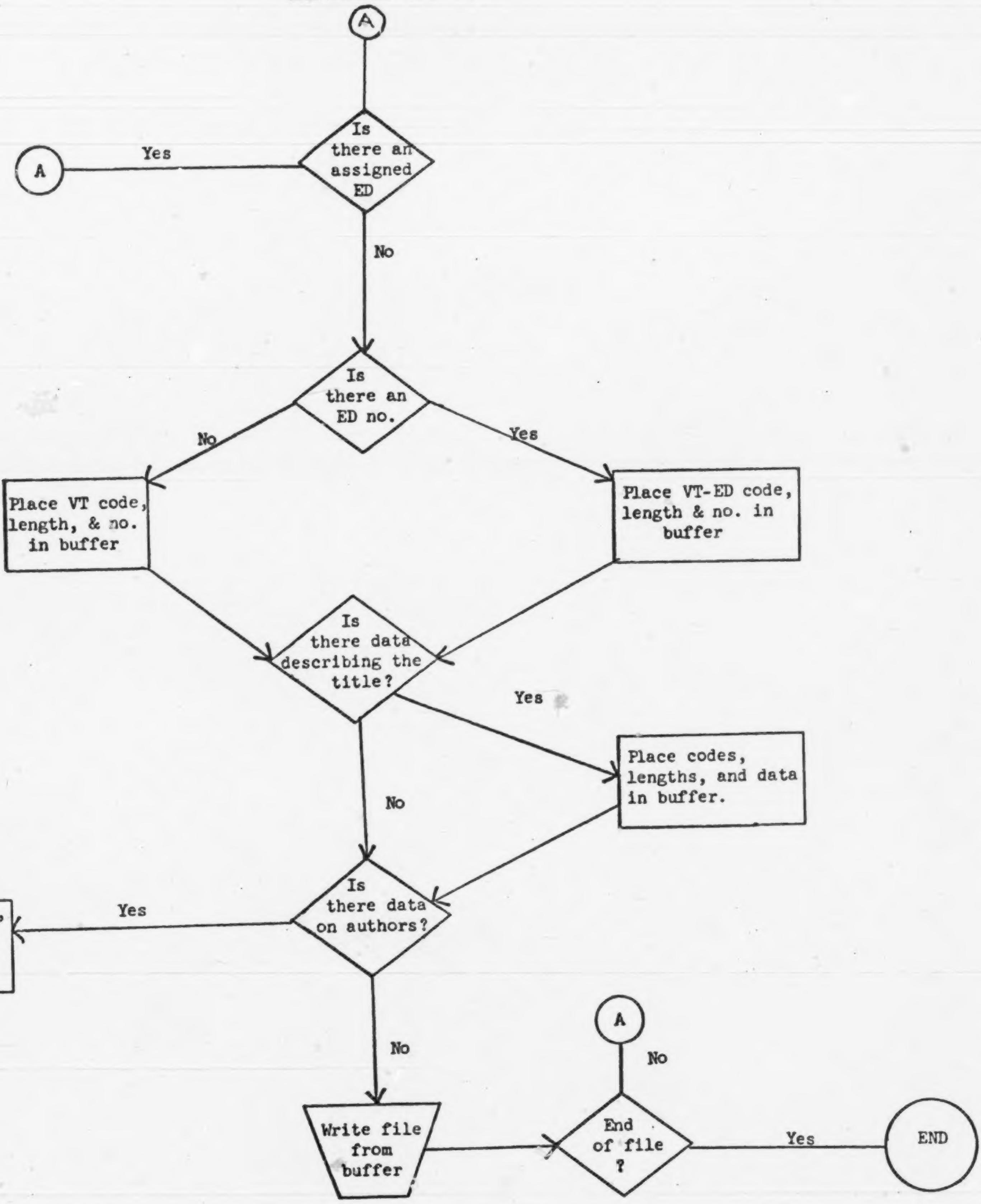

B
Go to EOJ1
EOJ2
END_CDE = 1
Yes
Go to CLOSE_FILES
No
End of cards
Yes
Gotto CLOSE_FILES
No
Move spaces to CARD_HOLD
Move INDEX-AREA to CARD_HOLD
Write tape from CARD_HOLD
Close files
Close CARDS DISC TAPE
END

RIE FORMAT

RIE Format


A
Read file
Substr =A?
No
Yes
Issue = ARM
Issue = AIM
Issue Q =?
=F
=W
=Sp
=Su
Issue = Fall
Issue = Winter
Issue = Spring
Issue = Summer
Get string with information on file
Get string with VT-ED numbers
Is there a personal author?
No
Yes
Get title
Get title and personal author
Are there any corporate authors
Yes
No
Get list of corporate authors
Are there any sponsoring agencies
Yes
No
Get list of sponsoring agencies
Get string with more information on file
A

RIE Format (cont'd.)


A
Is there an assigned ED
Yes
A
No
Is there an ED no.
No
Yes
Place VT code, length, & no. in buffer
Place VT-ED code, length & no. in buffer
Is there data describing the title?
Yes
Place codes, lengths, and data in buffer.
No
Is there data on authors?
Yes
Place codes, lengths and author data in buffer.
No
A
No
Write file from buffer
End of file ?
Yes
END

APPENDIX G

PAGE SAMPLES

	page
RESUME SECTION	265
QUARTERLY SUBJECT INDEX	268
QUARTERLY AUTHOR INDEX	271
QUARTERLY SUPPORTING SERVICE INDEX	274
QUARTERLY DOCUMENT NUMBER INDEX	277
CONVERSION OF DOCUMENT NUMBER INDEX	280
ANNUAL SUBJECT INDEX	282
ANNUAL AUTHOR INDEX	285
ANNUAL SUPPORTING SERVICE INDEX	288
ANNUAL DOCUMENT NUMBER INDEX	291

RESUME SECTION

6

VT 010 484
Siegfried, Larry R.
Source Unit in Feeds and Feeding.

Colorado State Univ., Ft. Collins. Agricultural Education Section
Colorado State Board for Community Colleges and Occupational Education, Denver
MF AVAILABLE IN VT-ERIC SET.
Agricultural Education Section, Department of Vocational Education, Colorado State University, Fort Collins, Colorado 80521 ($1.00).
PUB DATE - 69 41p.

DESCRIPTORS - *RESOURCE UNITS; RESOURCE MATERIALS; *UNIT PLAN; *VOCATIONAL AGRICULTURE; *AGRICULTURAL PRODUCTION; ANIMAL SCIENCE; *NUTRITION; LIVESTOCK; SECONDARY GRADES

ABSTRACT - Prepared by a graduate assistant in agricultural education, this source unit may be used by vocational agriculture teachers in planning and presenting a unit in feeds and feeding. The 13 sub-units, providing objectives, motivation techniques, study guides, references, and suggested teaching plans, cover: (1) nutrients necessary for livestock, (2) chemical composition and classification of feeds, (3) digestive systems of ruminants and non-ruminants, (4) digestion and adsorption of feeds by livestock, (5) feed preparation techniques, (6) role of proteins, carbohydrates, fats, minerals, and vitamins in livestock feeding, (7) feed additives, (8) balancing rations, and (9) computing lowest-cost balanced rations. (AW)

VT 010 527
Tree Identification: Western Pines.

California State Polytechnic Coll., San Luis Obispo. Vocational Education Productions
California State Dept. of Education, Sacramento. Vocational Education Section
MF AVAILABLE IN VT-ERIC SET.
Vocational Education Productions, California State Polytechnic College, San Luis Obispo, California 93401 ($2.00, Manual and 20 slides available for $6.50).
PUB DATE - 69 21p.

DESCRIPTORS - *GUIDES; *AGRICULTURAL EDUCATION; *IDENTIFICATION; *TREES; *FORESTRY; NATURAL RESOURCES; CLASSIFICATION; TAXONOMY

ABSTRACT - This multi-media kit, for use in agricultural education classes studying tree identification, consists of the following: (1) 10 slides of pines of commercial significance in the West, with the common and Latin names of the tree superimposed on a combined view of a typical specimen and a close-up view of the trunk and needles or leaves, (2) 10 slides identical to the ones identified, but without the names, to be used for testing purposes, (3) 10 transparency masters, one for each of the 10 species pictured on the slides, for use in producing overhead transparencies or other instructional materials, (4) short descriptions of the 10 pictured species, and (5) suggested identification procedures. The 10 Species of western pines covered are: (1) Douglas Fir, (2) Engelmann Spruce, (3) Idaho White Pine, (4) Incense Cedar, (5) Larch, (6) Lodgepole Pine, (7) Ponderosa Pine, (8) Sugar Pine, (9) Western Red Cedar, and (10) White Fir. The slide set may be obtained from the source of availability indicated for this document. (AW)

VT 010 528
Careers in Animal Industry.

California State Polytechnic Coll., San Luis Obispo. Vocational Education Productions
California State Dept. of Education, Sacramento. Vocational Education Section
DOCUMENT NOT AVAILABLE FROM EDRS.
Vocational Education Productions, California State Polytechnic College, San Luis Obispo, California 93401 ($.50; filmstrip, script, and taped narration $7.95).
PUB DATE - 69 15p.

DESCRIPTORS - *AUDIOVISUAL AIDS; *OCCUPATIONAL INFORMATION; *AGRICULTURAL OCCUPATIONS; *OFF FARM AGRICULTURAL OCCUPATIONS
IDENTIFIERS - *ANIMAL INDUSTRY

ABSTRACT - This illustrated companion script to a 50-frame filmstrip was developed by departments in the School of Agriculture, California State Polytechnic College to provide information on the history, progress, and variety of occupations in the animal industry. The film also discusses briefly the education needed and opportunities for advancement in animal industry occupations. (CH)

DESCRIPTORS - *TEACHING GUIDES; *DISTRIBUTIVE EDUCATION; *TEACHING TECHNIQUES; SALESMANSHIP; MERCHANDISING; COMMUNICATION SKILLS; CAREER PLANNING; SECONDARY GRADES

ABSTRACT - The teacher's guide is designed to aid the teacher-coordinator in planning individual and group activities for students in Distributive Education I and II. The teaching techniques suggested include case studies, structured reports, and assignments related to class study, the students' jobs, and club contests. Units containing suggested activities for Distributive Education I are: (1) School and Business Relationships, (2) Sales Development, (3) Business Communications, and (4) Organizations for Distribution. Units containing suggested activities for Distributive Education II are: (1) Career Planning for Professional Growth, (2) Economics for Distribution, (3) Merchandising, and (4) Business Psychology. Each unit contains teacher information and student forms for the suggested activities. (SB)

VT 002 683
Marketing I; Teachers Manual. Distributive Education Series.

North Carolina State Dept. of Public Instruction, Raleigh. Div. of Vocational Education
MF AVAILABLE IN VT-ERIC SET.
PUB DATE - ND 247p.

DESCRIPTORS - *DISTRIBUTIVE EDUCATION; *TEACHING GUIDES; *MARKETING; SALES OCCUPATIONS; SALESMANSHIP; MERCHANDISING; CAREER OPPORTUNITIES; GRADE 11

ABSTRACT - Developed by a committee of high school teacher-coordinators, this teacher's guide is for use in planning a first-year course in marketing for juniors who expect to enter some form of distribution, but who cannot, for various reasons, participate in a cooperative program of on-the-job training. A suggested time allotment and bibliography are provided for these units: (1) Introduction to Marketing I, (2) Sales Supporting Activities, (3) Salesmanship, (4) Sales Promotion, (5) Business Organization, and (6) Careers in Marketing. Each unit contains a list of materials and equipment needed, objectives, topical outline, and teaching suggestions. Supplementary forms and information for students are appended. A related document is available as VT 002 568 (also in this issue). (SB)

VT 010 256
Russell, Kenneth Lee
The How in Parliamentary Procedure.

Distributive Education Clubs of America
DOCUMENT NOT AVAILABLE FROM EDRS.
The Interstate Printers and Publishers, Inc., 19-27 North Jackson Street, Danville, Illinois 61832.
PUB DATE - 67 65p.

DESCRIPTORS - *DISTRIBUTIVE EDUCATION; *STUDENT ORGANIZATIONS; *CLASSROOM MATERIALS; *COCURRICULAR ACTIVITIES; *MANUALS
IDENTIFIERS - *PARLIAMENTARY PROCEDURES

ABSTRACT - This handbook was prepared by a professor of education for those who participate in group meetings. It is useful both to high school groups such as student councils and to adult civic groups. Topics discussed include (1) rule enforcement, (2) motions, (3) elections, (4) new business, and (5) requests. A student assignment book based on this manual is available as VT 010 257 (also in this issue). (JS)

VT 010 257
The How in Parliamentary Procedure, Student Assignments.

Texas Univ., Austin. Instructional Materials Lab.
DOCUMENT NOT AVAILABLE FROM EDRS.
Instructional Materials Laboratory, Distributive Education Department, Division of Extension, The University of Texas, Austin, Texas 78712.
PUB DATE - Jun69 29p.; Based on "The How in Parliamentary Procedure" by Kenneth L. Russell, The Interstate Printers and Publishers, Inc., Danville, Illinois, 1967.

DESCRIPTORS - *DISTRIBUTIVE EDUCATION; MANUALS; *STUDENT PROJECTS; *WORKBOOKS; *STUDY GUIDES; SECONDARY SCHOOLS; *CLASSROOM MATERIALS
IDENTIFIERS - *PARLIAMENTARY PROCEDURES

ABSTRACT - This student manual was published by a university materials laboratory to teach distributive education students the fundamentals of parliamentary procedure.

QUARTERLY SUBJECT INDEX

SUBJECT AND IDENTIFIER INDEX

PAGE

ABSTRACTS
VT 010 767 ED 037 582 81
Abstracts of Instructional Materials for Student Use in Vocational Cooperative Education.

ACCIDENT PREVENTION
VT 010 904 17
Farm Machinery Safety for the Institutional Farm.

ADMINISTRATOR GUIDES
VT 002 685 58
Trade Preparatory Training Planning Guide, Carpentry.
VT 010 615 22
Vocational Office Education in Florida High Schools.
VT 010 773 23
Administrative Handbook for Business and Distributive Education.
VT 011 216 ED 040 291 78
Applying the Cooperative Plan of Instruction to Manpower Programs. A Handbook for Supervisors and Coordinators.

ADULT EDUCATION
VT 010 245 21
Introduction to Automatic Data Processing.
VT 011 140 ED 039 365 24
Information Communication Occupations. A Suggested Curriculum Guide.
VT 011 169 26
Office Machines Resource Materials for Adult Programs in Business and Distributive Education.

ADULT FARMER EDUCATION
VT 002 841 6
Livestock Feeds and Feeding.

ADULT VOCATIONAL EDUCATION
VT 008 757 62
Welding and Flame-Cutting Processes and Practices.
VT 010 570 34
Franchise Merchandising. A Suggested Adult Course Outline.
VT 011 220 42
Oral Histology.

ADVERTISING LAYOUT
VT 010 412 32
Advertising Layout and Copy.

AGRICULTURAL CHEMICAL OCCUPATIONS
VT 001 346 5
Agricultural Chemicals.
VT 008 503 7
Agricultural Chemicals Employee.

PAGE

AGRICULTURAL EDUCATION
VT 000 382 3
Ornamental Horticulture for Vocational Agriculture in Alabama.
VT 007 073 7
A Handbook for Student Teachers and Supervising Teachers in Vocational Agriculture.
VT 010 362 9
Legal Principles for Farm Managers.
VT 010 363 9
Agricultural Cooperatives.
VT 010 527 12
Tree Identification: Western Pines.
VT 010 754 15
Fundamentals of Service: Power Trains.
VT 010 755 16
Fundamentals of Service: Hydraulics.
VT 010 756 16
Fundamentals of Service: Engines.
VT 010 757 16
Fundamentals of Service: Electrical Systems.
VT 010 968 ED 039 346 18
Farm Crop Production Technology: Field and Forage Crop and Fruit and Vine Production Options. A Suggested Curriculum.

AGRICULTURAL ENGINEERING
VT 001 571 5
Vocational Agriculture Course of Study Guide.
VT 010 251 7
Cold Metal Work in Vocational Agriculture.
VT 010 255 8
Overhead Projection; Transparency Masters for Vocational Agriculture (Including Supplements I-III).
VT 010 259 8
Course Guide for Basic Agricultural Mechanics.
VT 010 360 8
The Internal Combustion Engine.
VT 010 754 15
Fundamentals of Service: Power Trains.
VT 010 755 16
Fundamentals of Service: Hydraulics.
VT 010 756 16
Fundamentals of Service: Engines.
VT 010 757 16
Fundamentals of Service: Electrical Systems.
VT 010 765 ED 038 527 17
Agricultural Equipment Technology: A Suggested 2-Year Post High School Curriculum.

SUBJECT AND IDENTIFIER INDEX

PAGE

VT 010 235 69
Training Recommendations: Scaffolders.
VT 010 583 74
Building Maintenance, A Study Guide.

BUSINESS EDUCATION
VT 001 873 20
Education for Business.
VT 001 881 20
Bookkeeping I, Syllabus.
VT 010 114 20
Supplementary Materials Handbook for Intensive Business and Office Education.
VT 010 465 21
Data Processing, A Course of Study.
VT 010 570 34
Franchise Merchandising. A Suggested Adult Course Outline.
VT 010 773 23
Administrative Handbook for Business and Distributive Education.
VT 010 774 23
Bookkeeping Motivation Devices.
VT 010 775 23
Typewriting.
VT 010 776 24
Your Business Teaching.
VT 011 145 25
Typewriting for Business; Typewriting by a Whole (Gestalt) Method.

CALCULUS
VT 010 406 80
Calculus For Technicians.

CARBURETOR AIR CLEANERS
VT 010 638 76
Servicing Carburetor Air Cleaners.

CAREER GUIDANCE SERIES
VT 000 239 48
Industrial Arts Supplement.

CAREER PLANNING
VT 001 359 48
Industrial Arts.

CENTRAL SERVICE AIDES
VT 011 073 41
Guidelines for the Training of Central Service Aides.

CHILD CARE
VT 008 419 60
Nursery Assistants. Curriculum Guides in Trade and Industrial Education.

CHILD CARE OCCUPATIONS
VT 008 419 60
Nursery Assistants. Curriculum Guides in Trade and Industrial Education.

PAGE

CIVIL ENGINEERING
VT 002 645 56
Civil Engineering Technology, Field Book.
VT 002 656 56
Civil Engineering Technology.

CLASSROOM MATERIALS
VT 010 256 29
The How in Parliamentary Procedure.
VT 010 257 29
The How in Parliamentary Procedure, Student Assignments.
VT 011 143 25
Timed Writing, No. 1-7.
VT 011 145 25
Typewriting for Business; Typewriting by a Whole (Gestalt) Method.

CLOTHING INSTRUCTION
VT 008 421 61
Dressmaking.

COCURRICULAR ACTIVITIES
VT 010 256 29
The How in Parliamentary Procedure.

COLLEGE BOUND STUDENTS
VT 009 288 51
A Technical Drawing Course for College-Preparatory Students.

COMMUNICATION OCCUPATIONS
VT 011 140 ED 039 365 24
Information Communication Occupations. A Suggested Curriculum Guide.

COMMUNICATIONS
VT 002 661 58
Trade Preparatory Training Curriculum Guide, Graphics and Industrial Communications 1 and 2.

COMPUTER ORIENTED PROGRAMS
VT 010 458 32
Computer Games Management, Instructor's Manual.

COMPUTER SCIENCE
VT 010 709 81
Understanding Computers.

CONSERVATION EDUCATION
VT 000 949 4
Agricultural Production and Management.
VT 010 372 9
Directory of Conservation Education in Minnesota.

CONSUMER EDUCATION
VT 000 420 80
Teaching Consumer Education to Special Students on the Secondary Level.
VT 002 808 44
Dollars Demand Decisions.

QUARTERLY AUTHOR INDEX

PERSONAL AND INSTITUTIONAL AUTHOR INDEX

PAGE

AGRICULTURAL EDUCATION TEACHING MATERIALS CENTER, COLLEGE STATION, TEX.
VT 008 503 7
Agricultural Chemicals Employee.

AGRICULTURAL RESEARCH SERVICE (DOA), WASHINGTON, D.C. CONSUMER AND FOOD ECONOMICS RESEARCH DIV.
VT 011 294 46
A Guide to Budgeting for the Family.

AGRICULTURAL STABILIZATION AND CONSERVATION SERVICE (DOA), WASHINGTON, D.C.
VT 010 745 14
ASCS and You, A Teacher's Manual.
VT 010 746 15
ASCS and You, A Student Resource Unit.

ALABAMA STATE DEPT. OF EDUCATION, MONTGOMERY. AGRICULTURAL EDUCATION SERVICE
VT 000 378 2
A Specialized Course Outline for Advanced Vocational Agriculture Students in Forest Products Production Sales and Services.
VT 000 379 2
Forestry Production Sales and Services (A Student Study Guide.
VT 000 380 2
A Specialized Course Outline for Advanced Vocational Agriculture Students in Agricultural Supply, Sales and Services.
VT 000 381 3
Agriculture Supplies, Sales and Services (A Student Study Guide). Agribusiness Education Series.
VT 000 382 3
Ornamental Horticulture for Vocational Agriculture in Alabama.

ALABAMA UNIV., UNIVERSITY. DEPT. OF TRADE AND INDUSTRIAL EDUCATION
VT 010 582 74
Small Gasoline Engines; Study Guide and Job Tests.
VT 010 583 74
Building Maintenance, A Study Guide.
VT 010 590 74
Air Conditioning and Refrigeration.
VT 010 601 75
Volkswagen Mechanics, Study Guide and Job Tests.
VT 010 620 75
Everyday Arithmetic for Printers.
VT 010 621 75
Offset Printing, A Study Guide.
VT 010 627 76
Plumbing, A Study Guide.
VT 010 629 76
Weaving and Design Technology.

PAGE

VT 010 682 77
Textile Technician, A Study Guide.
VT 010 690 77
Automobile Body and Fender Repair and Refinishing, Job Tests.

ALBERTA DEPT. OF EDUCATION, EDMONTON
VT 010 439 53
Junior High School Curriculum Guide for Industrial Arts.

ALLEN, DELL K.
VT 002 608 56
Laboratory Manual for Experimental Metal-Cutting and Machine-Tool Evaluation.

ALLEY, WALTER
VT 010 263 70
How to Do the Complete Disc and Drum Brake Job.

AMERICAN ASSOCIATION FOR AGRICULTURAL ENGINEERING AND VOCATIONAL AGRICULTURE, ATHENS, GA.
VT 010 542 14
Tractor Fuels and Lubricants, Selecting and Storing.

AMERICAN DENTAL ASSISTANTS ASSOCIATION, CHICAGO, ILL. EDUCATION COMMITTEE
VT 011 220 42
Oral Histology.

AMERICAN SOCIETY OF RADIOLOGIC TECHNOLOGISTS, FOND DU LAC, WIS.
VT 007 363 50
Teaching Aids in Electricity.

AMMCO TOOLS INC., NORTH CHICAGO, ILL.
VT 010 268 71
Brake Service Manual.
VT 010 269 71
Disc Brake Manual.

AMMCO TOOLS, INC., NORTH CHICAGO, ILL.
VT 010 263 70
How to Do the Complete Disc and Drum Brake Job.
VT 010 267 70
Wheel Alignment Manual.

ARGYLE PUBLISHING CORP., NEW YORK, N.Y.
VT 011 042 41
Selected Medical Terminology.

ARIZONA UNIV., TUCSON. DEPT. OF AGRICULTURAL EDUCATION
VT 010 259 8
Course Guide for Basic Agricultural Mechanics.

PERSONAL AND INSTITUTIONAL AUTHOR INDEX

PAGE

ORANGE COUNTY BOARD OF EDUCATION, CALIF. EDUCATIONAL SERVICES DIV.
VT 000 420 80
Teaching Consumer Education to Special Students on the Secondary Level.

OREGON STATE BOARD OF EDUCATION, SALEM. DIV. OF COMMUNITY COLLEGES AND CAREER EDUCATION
VT 011 175 ED 040 285 41
Curriculum Guide for Health Occupations.

PEALE, BURMA ROSE, AND OTHERS
VT 000 420 80
Teaching Consumer Education to Special Students on the Secondary Level.

PENNSYLVANIA STATE UNIV., UNIVERSITY PARK. DEPT. OF AGRICULTURAL EDUCATION
VT 010 745 14
ASCS and You, A Teacher's Manual.
VT 010 746 15
ASCS and You, A Student Resource Unit.

PENNSYLVANIA YOUNG FARMERS ASSOCIATION
VT 010 362 9
Legal Principles for Farm Managers.

PENROD, L.O.
VT 010 620 75
Everyday Arithmetic for Printers.

PHIPPS, D. BART
VT 009 973 63
Basic Refrigeration for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 009 976 64
Sketching and Plan Reading for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 009 996 64
Fundamentals of Plumbing Installation for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 010 012 65
Industrial Blueprints for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 010 013 65
Basic Instruction for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 010 014 65
Drawing and Plan Reading for Apprentice Training in the Plumbing and Pipe Fitting Industry.

PAGE

VT 010 015 66
Rigging and Safety for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 010 026 67
Principles of Heating for Apprentice Training in the Plumbing and Pipe Fitting Industry.

PORTLAND STATE COLL., ORE.
VT 009 359 40
Nursing Home and Extended Care Administration.

POYNER, CLINTON B., SR., COMP.
VT 008 422 61
Food Trades. Curriculum Guides in Trade and Industrial Education.

PROGRAMS FOR ACHIEVEMENT IN READING, INC., PROVIDENCE, R.I.
VT 011 142 25
Typewriting for Business; Typewriting by a Whole (Gestalt) Method. Teacher's Manual and Key.
VT 011 143 25
Timed Writing, No. 1-7.
VT 011 145 25
Typewriting for Business; Typewriting by a Whole (Gestalt) Method.
VT 011 151 26
Phonics Review for Shorthand.
VT 011 152 26
Phonics Review for Shorthand. Teacher's Edition.

PRUDHOMME, RHETTA W., COMP.
VT 008 501 62
Household Operations. Curriculum Guides in Trade and Industrial Education.

PULASKI, RICHARD E.
VT 009 975 64
Theory and Practice of Natural Gas Installation for Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 010 024 66
Basic Instruction: Apprentice Training in the Plumbing and Pipe Fitting Industry.
VT 010 025 66
Basic Instruction Apprentice Training in the Plumbing and Pipe Fitting Industry.

RAWLS, BYRON F.
VT 000 378 2
A Specialized Course Outline for Advanced Vocational Agriculture Students in Forest Products Production Sales and Services.
VT 000 379 2
Forestry Production Sales and Services (A Student Study Guide.

QUARTERLY SUPPORTING SERVICES INDEX

VOCATIONAL AND SUPPORTING SERVICES INDEX

PAGE

AGRICULTURAL EDUCATION
VT 004 032 4
Supervised Occupational Experience Program in Vocational Agriculture for Arkansas, (Type C Program).
VT 004 591 282
A Pilot Program in New Teacher Preparation and In-Service Upgrading of Teachers and Supervisors.
VT 006 125 176
Evaluation of 4-H Veterinary Science Project in Three Counties of New York State.
VT 007 404 14
Guidelines for Program Development; High School Vocational Horticulture.
VT 007 476 ED 024 825 15
NEED FOR A POST-HIGH SCHOOL VOCATIONAL TRAINING PROGRAM IN HORTICULTURE.
VT 007 800 18
Adult Education in Agriculture.
VT 008 280 ED 029 134 97
DEVELOPING OCCUPATIONAL EXPERIENCE PROGRAMS IN AGRICULTURAL EDUCATION. BULL. 72H-5.
VT 008 368 131
Opportunities in Agricultural Education.
VT 008 427 132
Educational Needs and Employment Opportunities in Non-Farm Agriculture.
VT 008 458 ED 029 149 28
ONE-YEAR AND TWO-YEAR POST HIGH SCHOOL INSTITUTIONS WHICH OFFER PROGRAMS OF INSTRUCTION IN AGRICULTURE, 1968-69 DIRECTORY.
VT 008 561 133
South Dakota Agricultural Off-Farm Occupational Opportunities andTraining Needs.
VT 008 738 ED 030 738 35
A POLICY BULLETIN AND RECOMMENDATIONS FOR VOCATIONAL AGRICULTURE DEPARTMENTS IN WYOMING.
VT 008 800 192
A Study to Determine the Physical Facilities Needed for Vocational Agriculture.
VT 008 804 ED 030 747 146
REQUIREMENTS AND OPPORTUNITIES FOR ENTRY WORKERS IN THE OCCUPATION OF PRODUCING AND MARKETING ORNAMENTAL NURSERY AND GREENHOUSE SPECIALISTS.
VT 008 929 ED 030 761 197
HISTORICAL DEVELOPMENT OF AGRICULTURAL EDUCATION IN THE UNITED STATES PRIOR TO 1917. FINAL REPORT.
VT 008 957 237
The Commission on Education in Agriculture and Natural Resources.

PAGE

VT 008 968 ED 030 764 287
THE IDENTIFICATION OF OPINION LEADERS AMONG TEACHERS OF VOCATIONAL AGRICULTURE. FINAL REPORT. RESEARCH 40.
VT 008 996 ED 031 585 153
FACTORS AFFECTING THE ESTABLISHMENT OF YOUNG FARM OPERATORS IN IOWA AND IMPLICATIONS FOR AGRICULTURAL EDUCATION.
VT 009 782 290
Annual Report, Department of Agricultural Education, 1968-1969.
VT 009 920 ED 035 729 193
A TEACHING EXPERIMENT IN THE USE O GREENHOUSE FACILITIES IN VOCATIONAL AGRICULTURE.
VT 009 921 ED 035 730 300
AN EXPERIMENT IN TEACHING AGRICULTURAL OCCUPATIONS INFORMATION TO HIGH SCHOOL STUDENTS.
VT 009 990 ED 034 886 163
EDUCATION FOR OFF-FARM AGRICULTURAL OCCUPATIONS IN NEVADA.
VT 010 166 ED 034 901 73
AGRICULTURAL OCCUPATIONS PROGRAM DEVELOPMENT IN AREA VOCATIONAL SCHOOLS. FINAL REPORT.
VT 010 205 ED 035 761 74
AN EXPERIMENTAL GUIDE FOR PERSONNEL TRAINING REQUIREMENTS OF TECHNICIANS IN FUTURE FOOD IRRADIATION TECHNOLOGY INDUSTRIES. FINAL REPORT.
VT 010 280 ED 035 769 101
ATTITUDES OF SELECTED GROUPS CONCERNING THE ROLE OF VOCATIONAL-TECHNICAL EDUCATION PROGRAMS FOR OCCUPATIONS IN AGRICULTURAL RESOURCES.
VT 010 297 292
Farm Power Mechanics Competencies of Minnesota Vocational Agriculture Instructors.
VT 010 299 76
Vocational Horticulture Specialty Programs for Secondary Students.
VT 010 311 ED 035 773 102
AN EXPERIMENTAL EVALUATION OF APPROACHES TO PREPARING HIGH SCHOOL STUDENTS FOR AGRICULTURAL OCCUPATIONS OTHER THAN FARMING AND (PRINCIPLES VERSUS TRADITIONAL APPROACH TO TEACHING VOCATIONAL AGRICULTURE). FINAL REPORT.
VT 010 329 253
Research Studies in Progress in Agricultural Education in the North Atlantic Region, 1969-70.
VT 010 383 ED 038 504 77
DUTCH AGRICULTURAL EDUCATION.
VT 010 398 ED 036 647 186
THE RELEVANCE OF SECONDARY OCCUPATIONAL TRAINING IN AGRICULTURE TO OCCUPATIONAL PATTERNS AND IMAGES. FINAL REPORT.

VOCATIONAL AND SUPPORTING SERVICES INDEX

PAGE

VT 010 283 ED 036 633 223
Comprehensive Statewide Planning for Vocational Rehabilitation Services. Final Report, Volume II.

VT 010 284 ED 036 634 223
Comprehensive Statewide Planning for Vocational Rehabilitation Services. Final Report, Volume I.

VT 010 285 ED 037 553 185
Demographic Techniques for Manpower Planning in Developing Countries.

VT 010 286 ED 035 771 240
EDUCATION, HUMAN RESOURCES, AND DEVELOPMENT IN LATIN AMERICA.

VT 010 349 ED 039 323 224
Hard-Core Unemployment: A Selected, Annotated Bibliography.

VT 010 368 167
Employment and Manpower Problems in the Cities: Implications of the Report of the National Advisory Commission on Civil Disorders.

VT 010 495 ED 037 562 78
GOVERNOR'S MANPOWER CONFERENCE REPORT. 1969.

VT 010 541 ED 038 508 79
Summaries of Manpower Surveys and Reports for Developing Countries, 1958-68.

VT 010 585 ED 038 510 225
Poor People at Work, An Annotated Bibliography on Semi-professionals in Education, Health, and Welfare Services.

VT 010 608 80
Related Instruction for Apprentices.

VT 010 645 168
A Survey of the Manpower Needs in the Commercial Nursery Industry.

VT 010 658 168
Demand for Engineers and Technicians 1968.

VT 010 678 ED 039 329 225
Private Industry and the Disadvantaged Worker.

VT 010 708 ED 039 331 169
THE PRE-RETIREMENT YEARS: A LONGITUDINAL STUDY OF LABOR MARKET EXPERIENCE OF MEN. VOLUME TWO.

VT 010 718 169
Job Descriptions: How to Write and Use Them.

VT 010 731 ED 037 579 169
Health Manpower Source Book. Manpower Supply and Educational Statistics for Selected Health Occupations: 1968.

VT 010 753 ED 038 525 83
A STUDY OF MANPOWER DEVELOPMENT AND TRAINING ACT PROGRAMS IN NEW YORK STATE.

VT 010 800 ED 038 529 85
STRATEGIES FOR MEETING CORRECTIONAL TRAINING AND MANPOWER NEEDS, FOUR DEVELOPMENTAL PROJECTS.

PAGE

VT 010 806 ED 039 334 189
Cost-Benefit Analysis of Manpower Policies. Proceedings of a North American Conference.

VT 010 869 ED 039 338 170
JOBS AND CIVIL RIGHTS; THE ROLE OF THE FEDERAL GOVERNMENT IN PROMOTING EQUAL OPPORTUNITY IN EMPLOYMENT AND TRAINING. CLEARINGHOUSE PUBLICATION NO. 16.

VT 010 875 ED 039 340 170
GUIDE TO LOCAL OCCUPATIONAL INFORMATION.

VT 010 902 ED 039 342 227
EVALUATION OF THE MISSISSIPPI-IOWA FARM LABOR MOBILITY PROJECT.

OCCUPATIONAL PSYCHOLOGY

VT 007 108 176
Perceptual-Psychomotor Tests in Aircrew Selection.

VT 007 927 123
Nursing Attitudes and Turnover.

VT 007 988 ED 027 433 125
VOCATIONAL INTERESTS OF NON-PROFESSIONAL WOMEN. FINAL REPORT.

VT 008 002 266
Relationship of Vocational Satisfaction to the Correspondence of Job Reinforcement and Vocational Needs.

VT 008 512 ED 030 718 214
DEPENDENCY AND ITS IMPLICATIONS FOR REHABILITATION. NORTHEASTERN STUDIES IN VOCATIONAL REHABILITATION. MONOGRAPH NO. 1.

VT 008 798 ED 030 746 38
RESEARCH PROJECTS 1968.

OCCUPATIONAL SOCIOLOGY

VT 004 741 263
A Reconceptualization of Vocational Guidance.

VT 005 742 283
A Teacher's Role with the Disadvantaged.

VT 006 952 MP 000 610 264
Retraining Under the Manpower Development Training Act.

VT 007 663 ED 027 397 266
SOCIOLOGICAL STUDIES OF OCCUPATIONS; ABSTRACTS.

VT 008 003 125
Supervisor Perceptions of Occupational Environments and Roe's Classification of Occupations.

VT 008 171 ED 029 123 129
IN SEARCH OF OPPORTUNITY. A STUDY OF POST HIGH SCHOOL MIGRATION IN MINNESOTA. TECHNICAL BULLETIN 247.

VT 008 578 134
They Carry the Burden Alone; The Socio-Economic Living Pattern of Oregon Women with Dependents.

VT 008 580 ED 029 981 135
THE MODEL NEIGHBORHOOD JOBSEEKER. A SURVEY OF HAWAII STATE EMPLOYMENT SERVICE JOB APPLICANTS.

QUARTERLY DOCUMENT NUMBER INDEX

DOCUMENT NUMBER INDEX

PAGE

VT 000 013 28
Let's Analyze and Sell.

VT 000 239 48
Industrial Arts Supplement.

VT 000 270 40
Manual for Instructors of Vocational Nursing, Pre-Clinical Period, 16-20 Weeks.

VT 000 282 2
A Training Program for Vocational Agriculture in Wildlife and Recreation.

VT 000 330 40
Instructor's Source Book, Vocational Nurse Training Program.

VT 000 378 2
A Specialized Course Outline for Advanced Vocational Agriculture Students in Forest Products Production Sales and Services.

VT 000 379 2
Forestry Production Sales and Services (A Student Study Guide.

VT 000 380 2
A Specialized Course Outline for Advanced Vocational Agriculture Students in Agricultural Supply, Sales and Services.

VT 000 381 3
Agriculture Supplies, Sales and Services (A Student Study Guide). Agribusiness Education Series.

VT 000 382 3
Ornamental Horticulture for Vocational Agriculture in Alabama.

VT 000 420 80
Teaching Consumer Education to Special Students on the Secondary Level.

VT 000 946 4
Agriculture II: A Curriculum Unit Outline for Vocational Agriculture.

VT 000 949 4
Agricultural Production and Management.

VT 000 989 4
Agricultural Instruction, Units I-VIII.

VT 001 093 4
Course of Study for Ornamental Horticulture Courses.

PAGE

VT 001 238 5
Source Units for the Dairy Enterprise.

VT 001 323 28
Test Construction in Distributive Education.

VT 001 346 5
Agricultural Chemicals.

VT 001 359 48
Industrial Arts.

VT 001 571 5
Vocational Agriculture Course of Study Guide.

VT 001 873 20
Education for Business.

VT 001 881 20
Bookkeeping I, Syllabus.

VT 001 998 44
Home Economics Humanities Media Guide.

VT 002 053 56
Water and Sewerage, A Course of Study.

VT 002 112 48
A Guide for Teaching Power Mechanics in Industrial Arts in Indiana.

VT 002 345 49
Industrial Arts Safety Instruction.

VT 002 568 28
Marketing II, Teachers Manual.

VT 002 588 28
Cooperative Distributive Education; Teacher's Guide to Student Activities.

VT 002 608 56
Laboratory Manual for Experimental Metal-Cutting and Machine-Tool Evaluation.

VT 002 645 56
Civil Engineering Technology, Field Book.

VT 002 650 49
Guide Lines for the Junior High School Industrial Arts Program.

VT 002 656 56
Civil Engineering Technology.

DOCUMENT NUMBER INDEX

PAGE

VT 010 478 Related Guide..Job Employment, Distributive Education. 33

VT 010 480 Source Unit in Animal Breeding. 11

VT 010 484 Source Unit in Feeds and Feeding. 12

VT 010 505 Modern Metalworking Technology. 53

VT 010 527 Tree Identification: Western Pines. 12

VT 010 528 Careers in Animal Industry. 12

VT 010 529 Careers in Government Service. 81

VT 010 530 Careers in Natural Resources Management. 81

VT 010 531 Careers in Ornamental Horticulture. 13

VT 010 532 Careers in Farm Supplies. 13

VT 010 533 Careers in Farm Services. 13

VT 010 534 Careers in Farm Machinery Sales and Service. 13

VT 010 535 Careers in Crops. 14

VT 010 542 Tractor Fuels and Lubricants, Selecting and Storing. 14

VT 010 543 What a Fireman Should Know About Sprinkler Systems. 72

VT 010 544 Fire Department Driver Training Manual of Instruction. 72

VT 010 545 Manual of Salvage Operations. 72

VT 010 546 Recommendation 19: The Training of Power Station Operators Coal, Oil and Gas Fired Stations. 73

VT 010 570 Franchise Merchandising. A Suggested Adult Course Outline. 34

PAGE

VT 010 571 California Fire Training Program; Book 2 of Block III, Use of Fire Apparatus and Equipment. 73

VT 010 572 California Fire Training Program; Book 1 of Block III, Use of Fire Apparatus and Equipment. 73

VT 010 582 Small Gasoline Engines; Study Guide and Job Tests. 74

VT 010 583 Building Maintenance, A Study Guide. 74

VT 010 590 Air Conditioning and Refrigeration. 74

VT 010 601 Volkswagen Mechanics, Study Guide and Job Tests. 75

VT 010 611 Business and Office Education: Before and After in Indiana. 22

VT 010 615 Vocational Office Education in Florida High Schools. 22

VT 010 619 Variety Store Merchandising. 34

VT 010 620 Everyday Arithmetic for Printers. 75

VT 010 621 Offset Printing, A Study Guide. 75

VT 010 627 Plumbing, A Study Guide. 76

VT 010 629 Weaving and Design Technology. 76

VT 010 638 Servicing Carburetor Air Cleaners. 76

VT 010 682 Textile Technician, A Study Guide. 77

VT 010 690 Automobile Body and Fender Repair and Refinishing, Job Tests. 77

VT 010 709 Understanding Computers. 81

VT 010 726 Nitrates in Water Supplies, Field Crops, and Ruminant Nutrition. 14

VT 010 734 Introduction to Programming PDP-8 Family Computers. 22

CONVERSION OF DOCUMENT NUMBER

INDEX

CONVERSION OF DOCUMENT NUMBER INDEX[1]

VT NUMBERS CONVERTED TO ED NUMBERS OR MP NUMBERS[2]

VT 002 431	ED 035 715	VT 008 273	ED 029 964	VT 008 804	ED 030 747
VT 002 508	ED 029 943	VT 008 280	ED 029 134	VT 008 815	ED 039 316
VT 004 667	ED 038 488	VT 008 303	ED 032 395	VT 008 830	ED 030 750
VT 006 275	ED 025 609	VT 008 429	ED 029 147	VT 008 831	ED 030 751
VT 006 277	ED 025 611	VT 008 433	ED 029 148	VT 008 832	ED 030 752
VT 006 285	ED 035 788	VT 008 438	ED 029 970	VT 008 833	ED 030 753
VT 006 952	MP 000 610	VT 008 444	ED 030 716	VT 008 844	ED 031 574
VT 007 147	ED 025 632	VT 008 451	ED 028 306	VT 008 850	ED 031 577
VT 007 154	ED 023 896	VT 008 458	ED 029 149	VT 008 854	ED 032 407
VT 007 211	ED 032 390	VT 008 511	ED 030 717	VT 008 860	ED 030 756
VT 007 315	ED 027 388	VT 008 512	ED 030 718	VT 008 863	ED 031 578
VT 007 476	ED 024 825	VT 008 513	ED 029 976	VT 008 868	ED 036 618
VT 007 547	ED 026 517	VT 008 538	ED 038 489	VT 008 874	ED 030 757
VT 007 555	ED 025 654	VT 008 539	ED 038 490	VT 008 899	ED 032 410
VT 007 560	ED 033 188	VT 008 544	ED 029 977	VT 008 902	ED 030 759
VT 007 582	ED 025 658	VT 008 547	ED 030 722	VT 008 906	ED 031 579
VT 007 662	ED 037 522	VT 008 554	ED 029 978	VT 008 921	ED 030 760
VT 007 663	ED 027 397	VT 008 576	ED 031 566	VT 008 926	ED 031 580
VT 007 676	ED 025 672	VT 008 579	ED 030 723	VT 008 929	ED 030 761
VT 007 677	ED 025 673	VT 008 580	ED 029 981	VT 008 930	ED 033 199
VT 007 747	ED 034 019	VT 008 592	ED 030 724	VT 008 947	ED 032 073
VT 007 776	ED 026 531	VT 008 593	ED 030 725	VT 008 951	ED 032 413
VT 007 790	ED 025 679	VT 008 596	ED 032 397	VT 008 952	ED 030 762
VT 007 797	ED 029 103	VT 008 600	ED 029 989	VT 008 953	ED 030 763
VT 007 805	ED 039 312	VT 008 601	ED 029 990	VT 008 968	ED 030 764
VT 007 831	ED 029 106	VT 008 607	ED 030 726	VT 008 970	ED 030 765
VT 007 881	ED 029 956	VT 008 609	ED 031 568	VT 008 971	ED 032 414
VT 007 883	ED 026 541	VT 008 631	ED 029 993	VT 008 974	ED 030 766
VT 007 887	ED 026 542	VT 008 632	ED 029 994	VT 008 976	ED 030 767
VT 007 920	ED 036 613	VT 008 635	ED 033 196	VT 008 980	ED 031 583
VT 007 931	ED 027 425	VT 008 647	ED 030 729	VT 008 982	ED 030 768
VT 007 971	ED 039 313	VT 008 693	ED 030 731	VT 008 996	ED 031 585
VT 007 988	ED 027 433	VT 008 707	ED 031 569	VT 009 016	ED 031 588
VT 008 008	ED 029 112	VT 008 718	ED 033 197	VT 009 017	ED 031 589
VT 008 051	ED 029 [illegible]6	VT 008 719	ED 031 570	VT 009 019	ED 032 415
VT 008 082	ED 028 288	VT 008 727	ED 032 398	VT 009 057	ED 031 593
VT 008 091	ED 032 392	VT 008 728	ED 032 399	VT 009 075	ED 033 200
VT 008 127	ED 028 292	VT 008 729	ED 032 400	VT 009 082	ED 033 201
VT 008 129	ED 029 121	VT 008 730	ED 032 401	VT 009 086	ED 031 595
VT 008 138	ED 029 122	VT 008 735	ED 030 736	VT 009 087	ED 031 596
VT 008 146	ED 032 393	VT 008 736	ED 030 737	VT 009 126	ED 033 202
VT 008 150	ED 028 294	VT 008 738	ED 030 738	VT 009 154	ED 032 419
VT 008 171	ED 029 123	VT 008 750	ED 036 616	VT 009 158	ED 031 600
VT 008 173	ED 028 297	VT 008 753	ED 032 402	VT 009 159	ED 033 204
VT 008 177	ED 032 394	VT 008 769	ED 032 403	VT 009 185	ED 031 601
VT 008 182	ED 029 124	VT 008 770	ED 030 743	VT 009 188	ED 031 602
VT 008 183	ED 028 298	VT 008 776	ED 032 405	VT 009 190	ED 032 420
VT 008 185	ED 031 562	VT 008 777	ED 032 406	VT 009 192	ED 031 603
VT 008 198	ED 029 126	VT 008 787	ED 031 572	VT 009 196	ED 033 209
VT 008 222	ED 029 130	VT 008 798	ED 030 746	VT 009 198	ED 032 421

---

[1] This index is cumulated annually beginning with the Fall issue of ARM.

[2] Any document with an MP number is available from EDRS by using the appropriate MP number.

ANNUAL SUBJECT INDEX

SUBJECT AND IDENTIFIER INDEX

ABRAHAM JACOBS MEMORIAL SYMPOSIUM
66: 5436

ABSENTEEISM
66: 3752

ABSTRACTS
66: 7309
67: 4817
68: 7262, 7466, 7618, 7619,
 7620, 7621
 8602
69: 8588
 10149, 10295, 10322, 10328

AC GENERATORS
63: 8806

ACADEMIC ABILITY
68: 6571

ACADEMIC ACHIEVEMENT
60: 4618
66: 3314
 4576
67: 3758
68: 8151
69: 8373

ACADEMIC APTITUDE
68: 6600
69: 8277

ACADEMIC ASPIRATION
67: 7266, 7530
68: 6571

ACADEMIC EDUCATION
67: 6814
68: 7660

ACCELERATED COURSES
63: 2539

ACCELERATED PROGRAMS
65: 4508

ACCELERATED VOCATIONAL TRAINING
65: 4508

ACCIDENT PREVENTION
nd: 9942
63: 150
64: 1678
66: 10041
68: 8071, 8724

ACCOUNTANTS
68: 9363, 9364

ACCOUNTING
nd: 8060
65: 9229
68: 5955

ACCREDITATION (INSTITUTIONS)
64: 1532, 1900
65: 4652
 5997
 6027
66: 2638
 6254
67: 4746
 5999
 6000
 7821
68: 7497, 7819
 9231, 9232, 9237

ACHIEVEMENT
68: 6862
 7504

ACHIEVEMENT GAINS
68: 6671

ACHIEVEMENT RATING
69: 8648

ACHIEVEMENT TESTS
63: 5237
 8276
64: 5560
68: 7224
 8505
69: 9610

ACTION PROGRAMS (COMMUNITY)
67: 6786

ACTIVITIES
67: 4996

ADHESIVES
68: 7858

ADMINISTRATION
67: 4792

ADMINISTRATIVE CHANGE
68: 5870

ADMINISTRATIVE ORGANIZATION
nd: 5483
66: 2553
 6691
67: 6573
69: 8391

ADMINISTRATIVE PERSONNEL
66: 877
68: 7372, 7679
69: 7837, 7970

ADMINISTRATIVE POLICY
nd: 5483
63: 2028, 2383
64: 1521
 6462
66: 6348
67: 3615
 5316
 7084
68: 8659

ADMINISTRATIVE PROBLEMS
65: 6616
68: 7079, 7900
69: 9839
 10456

ADMINISTRATOR ATTITUDES
nd: 8739
65: 4575
67: 7067
68: 6677, 6769
 7322, 7566, 7693

ADMINISTRATOR BACKGROUND
69: 9839

ADMINISTRATOR CHARACTERISTICS
69: 9839

ADMINISTRATOR GUIDES
nd: 1463
 2641
63: 1702
 4121
64: 2496
 5615
65: 1606
 2132, 2597
66: 1319, 1509, 1552, 1673
 2504, 2592, 2593
 5283
 7326
67: 6657
68: 7787
 8050, 8062
69: 8802

ADMINISTRATOR GUIDES (CONTINUED)
69: 9894, 9899
 10511

ADMINISTRATOR RESPONSIBILITY
nd: 1463
68: 5874

ADMINISTRATOR ROLE
nd: 1463
67: 2727
68: 5874

ADMISSION (SCHOOL)
68: 7365

ADMISSION CRITERIA
65: 2541
66: 1897
 4019
68: 6686
 7582

ADOLESCENTS
67: 8522

ADOPTION (IDEAS)
67: 9843
68: 6014

ADULT BASIC EDUCATION
64: 1555
66: 8653

ADULT CHARACTERISTICS
68: 6506

ADULT EDUCATION
nd: 7176
65: 1695
 6426
68: 6720
 7741
69: 9491
 10558

ADULT EDUCATION PROGRAMS
62: 10102
66: 10103, 10104, 10105, 10108,
 10121, 10122, 10123
67: 10106, 10107, 10120
68: 7059

ADULT FARMER EDUCATION
nd: 544
65: 2386
 8061
 9252
66: 4585
 6823
68: 5855
 8646, 8656
69: 10098, 10426, 10506

ADULT LEARNING
64: 4535

ADULT STUDENTS
68: 7059

ADULT VOCATIONAL EDUCATION
nd: 5579
 8418, 8573, 8910, 8911
63: 831
 2013
 8482
64: 2909
 3608
 9047, 9090
65: 3901
 4508
 8959, 8960, 8961, 8985,
 8986
 9001, 9002, 9003, 9023,
 9024, 9025, 9026, 9027,

AIR CONDITIONING (CONTINUED)
68: 8342

AIRCRAFT PILOTS
67: 6804
69: 9490, 9582, 9588

ALGEBRA
69: 9819

ALLIED HEALTH PROFESSIONS PROJECTS
68: 7625

AMA ANNUAL CONFERENCE ON MEDICAL EDUCATION, CHICAGO, ILLINOIS, FEBRUARY 9, 1969
69: 8641

AMERICAN ASSOCIATION OF DENTAL SCHOOLS
68: 7274

AMERICAN ASSOCIATION OF JUNIOR COLLEGES, EAST LANSING, MICHIGAN, FEBRUARY 21, 1969
69: 8556

AMERICAN ASSOCIATION OF TEACHER EDUCATORS IN AGRICULTURE
68: 7696

AMERICAN DENTAL ASSOCIATION
67: 5316

AMERICAN HISTORY
66: 1914

AMERICAN INDIANS
68: 6999
8791

AMERICAN INDUSTRIAL ARTS ASSOCIATION
69: 9853

AMERICAN INDUSTRIAL ARTS STUDENT ASSOCIATION
67: 1954

AMERICAN INDUSTRIAL TECHNICAL EDUCATION CONFERENCE, COLUMBUS, OHIO, JUNE 20-21, 1967
67: 6123

AMERICAN INDUSTRY PROJECT
65: 2861

AMERICAN SOCIETY FOR ENGINEERING EDUCATION, UNIVERSITY OF CALIFORNIA, LOS ANGELES, JUNE 17-20, 1968
68: 7247

AMERICAN SOCIETY TOOL AND MANUFACTURING ENGINEERS
68: 8762

AMERICAN STATISTICAL ASSOCIATION AND ALLIED SOCIETIES
67: 6840

AMERICAN VOCATIONAL ASSOCIATION
68: 7380

AMERICAN VOCATIONAL ASSOCIATION CONVENTION, DALLAS, TEXAS, DECEMBER 9-13, 1968
68: 8586

AMERICAN VOCATIONAL ASSOCIATION CONVENTION, DALLAS, TEXAS, 1968
68: 8643

AMPLIFIERS
66: 10104

ANATOMY
64: 9326

ANATOMY (CONTINUED)
65: 9308, 9678

ANIMAL CARETAKERS
nd: 9320, 9327

ANIMAL SCIENCE
nd: 9325
10424
63: 1400
65: 57
66: 9775
68: 8764, 8766, 8767, 8771,
8822, 8933, 8934, 8935,
8936, 8958
69: 7993
8765, 8949
9629, 9871, 9872

ANNOTATED BIBLIOGRAPHIES
62: 6412
64: 6325
65: 2555
3782
66: 2721
4196, 4684, 4978
5225
6465
7699
67: 2389
3779
6331, 6810
8083, 8880
68: 5631
7118, 7228, 7234, 7239,
7262, 7420, 7466, 7618,
7619, 7620, 7621, 7903
8117, 8577, 8888
69: 7889
8568, 8591
9006, 9007, 9055, 9352,
9404, 9432, 9563, 9622,
9627, 9857, 9936
10077, 10295, 10322, 10328,
10364, 10430, 10464, 10558
70: 10509, 10829
11173, 11200

ANNUAL AMERICAN VOCATIONAL ASSOCIATION CONVENTION (63RD, BOSTON, MASSACHUSETTS, DECEMBER 6-9, 1969
69: 10450

ANNUAL AMERICAN VOCATIONAL ASSOCIATION CONVENTION (63RD, BOSTON, MASSACHUSETTS, DECEMBER 6-9, 1969)
65: 10443
69: 10110, 10111, 10174, 10176,
10298, 10302, 10305, 10307,
10392, 10393, 10394, 10413,
10414, 10415, 10416, 10431,
10432, 10433, 10434, 10435,
10440, 10441, 10444, 10445,
10446, 10447, 10448, 10449,
10451, 10452, 10453, 10454,
10456

ANNUAL AMERICAN VOCATIONAL ASSOCIATION CONVENTION (63RD, BOSTON, MASSACHUSETTS, DECEMBER 6-9, 1969).
69: 10455

ANNUAL CONFERENCE ON INDUSTRIAL EDUCATION, ANAHEIM, CALIFORNIA, MARCH 4-6, 1964
64: 4611

ANNUAL CONFERENCE ON INDUSTRIAL EDUCATION, SACRAMENTO, CALIFORNIA, MARCH 14-16, 1967
67: 2214

ANNUAL CONVENTION OF THE NATIONAL ASSOCIATION OF PSYCHIATRIC TECHNOLOGY, FRESNO, CALIFORNIA, OCTOBER, 1967
nd: 10316

ANNUAL REPORTS
65: 2109
66: 1367, 1621
67: 587
2410, 2428
5253, 5316
68: 5462
7622, 7741
8054
69: 9991

ANNUAL RESEARCH CONFERENCE
67: 6624

ANSWER KEYS
68: 7958

ANXIETY
68: 6669

APPRENTICESHIPS
nd: 8610, 8810, 8811, 8812
53: 9875
62: 717
63: 4800
64: 774
3524, 3808
65: 4408, 4600, 4798
66: 1367, 1552
3245, 3280, 3554
4799, 4801, 4876
8415, 8651
67: 3595
4712, 4803
7084
8841
9374
10222, 10224, 10225
68: 7414
8374, 8569, 8633, 8676,
8838
9022, 9366, 9367, 9368,
9371, 9372, 9373, 9376,
9378, 9379
10155, 10220, 10221, 10226,
10227, 10234, 10236
69: 9377
10214

APTITUDE
67: 2362

APTITUDE TESTS
67: 5978
69: 7817
8629

APTITUDES
67: 7044

ARCHITECTURAL EDUCATION
69: 10769

AREA VOCATIONAL SCHOOLS
nd: 8244
63: 1702
5178
64: 4675, 4679
65: 2359
4960
66: 1509, 1718, 1827
4666
67: 1992
4805
68: 5860
7482
8015, 8390, 8392
69: 8389, 8391, 8629
9169

ARIZONA
64: 3608

ANNUAL AUTHOR INDEX

PERSONAL AND INSTITUTIONAL AUTHOR INDEX

ABILENE PUBLIC SCHOOLS, TEX.
65: 352

ABRAMS, HARVEY A.
68: 8775

ACHIEVEMENT MOTIVATION DEVELOPMENT PROJECT, CAMBRIDGE, MASS.
69: 8366

ADAMS, JERRY
nd: 8250

ADAMS, JON P.
69: 7998

ADAMS, WALTER, ED.
67: 2690

ADKINS, DOLORES, AND OTHERS
65: 107

ADVISORY COUNCIL ON VOCATIONAL EDUCATION, WASHINGTON, D.C.
68: 7877

AEROSPACE MEDICAL RESEARCH LABS., WRIGHT-PATTERSON AFB, OHIO
67: 7098

AEROSPACE MEDICAL RESEARCH LABS, WRIGHT PATTERSON AFB, OHIO.
67: 7099

AEROSPACE MEDICAL RESEARCH LABS, WRIGHT-PATTERSON AFB, OHIO
67: 7101

AGAN, R.J., AND OTHERS
63: 2213

AGAN, RAYMOND J.
68: 6931

AGENCY FOR INTERNATIONAL DEVELOPMENT, WASHINGTON, D.C. COMMUNICATIONS RESOURCES DIV.
62: 2733

AGENCY FOR INTERNATIONAL DEVELOPMENT, WASHINGTON, D.C. CRD-PUBLICATIONS AND TECHNICAL SERVICES BRANCH
63: 2664

AGENCY FOR INTERNATIONAL DEVELOPMENT, WASHINGTON, D.C. OFFICE OF LABOR AFFAIRS
66: 5478

AGRICULTURAL EDUCATION TEACHING MATERIALS CENTER, COLLEGE STATION, TEX.
68: 8252

AGRICULTURAL RESEARCH SERVICE (DOA), WASHINGTON, D.C.
65: 9943
67: 10404

AIKEN, MICHAEL
67: 5475

AIKIN, BRENTON R.
63: 3691

AINSWORTH, ROBERT G.
67: 6856

AIR-CONDITIONING AND REFRIGERATION INST., ARLINGTON, VA.
68: 5853

AIR-CONDITIONING AND REFRIGERATION INST., ARLINGTON, VA. (CONTINUED)
68: 7607

AKAMINE, TOSHIO
68: 6919

AKRON PUBLIC SCHOOLS, OHIO
64: 445, 446, 447

ALABAMA RESEARCH COORDINATING UNIT FOR VOCATIONAL AND TECHNICAL EDUCATION, MONTGOMERY
69: 9791

ALABAMA STATE DEPT. OF EDUCATION, MONTGOMERY
nd: 176

ALABAMA STATE DEPT. OF EDUCATION, MONTGOMERY. AGRICULTURAL EDUCATION SERVICE
66: 367, 368, 372, 373,
374, 375, 376, 377,
383

ALABAMA STATE DEPT. OF EDUCATION, MONTGOMERY. TRADE AND INDUSTRIAL EDUCATION
67: 10589

ALABAMA STATE EMPLOYMENT SERVICE, MONTGOMERY
68: 7503

ALABAMA UNIV., UNIVERSITY DEPT. OF TRADE AND INDUSTRIAL EDUCATION
67: 10589

ALABAMA UNIV., UNIVERSITY. DEPT. OF TRADE AND INDUSTRIAL EDUCATION
68: 10581,10691
69: 10592,10687

ALAMEDA COUNTY HEALTH DEPT., OAKLAND, CALIF.
68: 7381

ALASKA OFFICE OF THE GOVERNOR, JUNEAU
65: 2511

ALASKA STATE DEPT. OF EDUCATION, JUNEAU. DIV. OF VOCATIONAL EDUCATION
66: 5283
8640

ALBORN A. AND M. COLL., LORMAN, MISS.
69: 8277

ALBRACHT, JAMES
68: 7980

ALCORN A AND M COLL., LORMAN, MISS. DIV. OF VOCATIONAL EDUCATION
65: 4558

ALLEN, DAVID
64: 4211, 4302
67: 4946

ALLEN, DAVID, AND OTHERS
65: 1916

ALLEN, DWIGHT W.
67: 3229
68: 7532

ALLEN, MARY P.
68: 7180

ALLEN, RAY
65: 9310
68: 9311

ALLIED HEALTH PROFESSIONS PROJECTS, LOS ANGELES, CALIF.
68: 7625, 7841

ALLINGTON CORP., ALEXANDRIA, VA.
68: 6188

ALSCHULER, ALFRED, AND OTHERS
69: 8366, 8373

ALTMAN, CHARLES DWAYNE
68: 7809

ALTOONA AREA HIGH SCHOOL, PA.
69: 8794

AMARILLO COLL., TEX.
68: 7422

AMBERSON, MAX L., AND OTHERS
69: 10456

AMELON, DONALD JAMES
69: 9184

AMERICAN ASSOCIATION FOR AGRICULTURAL ENGINEERING AND VOCATIONAL AGRICULTURE, ATHENS, GA.
69: 9981, 9982
10426

AMERICAN ASSOCIATION OF DENTAL EXAMINERS
67: 7091

AMERICAN ASSOCIATION OF DENTAL EXAMINERS, ALBANY, N.Y.
69: 6421

AMERICAN ASSOCIATION OF DENTAL SCHOOLS, WASHINGTON, D.C.
68: 7274

AMERICAN ASSOCIATION OF JUNIOR COLLEGES, WASHINGTON, D.C.
nd: 7136
66: 8063
68: 5467
7694

AMERICAN ASSOCIATION OF MEDICAL RECORD LIBRARIANS, CHICAGO, ILL.
65: 5997

AMERICAN ASSOCIATION OF TEACHER EDUCATORS IN AGRICULTURE, ST. PAUL, MINN.
69: 10456

AMERICAN COLLEGE OF DENTISTS
68: 7104

AMERICAN COUNCIL OF INDUSTRIAL ARTS SUPERVISORS
69: 7808

AMERICAN COUNCIL ON EDUCATION, WASHINGTON, D.C.
65: 2236

AMERICAN COUNCIL ON INDUSTRIAL ARTS TEACHER EDUCATION, WASHINGTON, D.C.
67: 2091

AMERICAN DENTAL ASSOCIATION, CHICAGO, ILL.
64: 1532, 1900
66: 2143

AMERICAN DENTAL ASSOCIATION,
CHICAGO, ILL. (CONTINUED)
67: 5316
68: 9887

AMERICAN DENTAL ASSOCIATION,
CHICAGO, ILL. COUNCIL ON DENTAL
EDUCATION
65: 4652
66: 1964

AMERICAN DENTAL ASSOCIATION,
CHICAGO, ILL. DIV. OF EDUCATIONAL
MEASUREMENTS
69: 10058

AMERICAN ENTERPRISE INST. FOR
PUBLIC POLICY RESEARCH,
WASHINGTON, D.C.
63: 4359

AMERICAN FOUNDATION FOR THE
BLIND, NEW YORK, N.Y.
67: 4249

AMERICAN FOUNDATION ON AUTOMATION
AND EMPLOYMENT, INC., NEW YORK,
N.Y.
69: 10065

AMERICAN HOME ECONOMICS
ASSOCIATION, WASHINGTON, D.C.
67: 4817
8083
68: 7114, 7262
69: 9352

AMERICAN HOSPITAL ASSOCIATION,
CHICAGO, ILL.
67: 5162

AMERICAN INDUSTRIAL ARTS
ASSOCIATION, WASHINGTON, D.C.
nd: 5740
67: 1954
5837
69: 7808
9853

AMERICAN INDUSTRIAL ARTS
ASSOCIATION, WASHINGTON, D.C.
INTERNATIONAL RELATIONS COMMITTEE
69: 9277

AMERICAN INST. FOR RESEARCH IN
THE BEHAVIORAL SCIENCES, PALO
ALTO, CALIF.
68: 8814

AMERICAN INSTITUTES FOR RESEARCH,
PITTSBURGH, PA.
nd: 8526, 8527
65: 1394
66: 1395, 1396, 1397
67: 4848
68: 6859
69: 8451, 8516, 8588

AMERICAN IRON AND STEEL INST.,
NEW YORK, N.Y.
66: 4684

AMERICAN MEDICAL ASSOCIATION,
CHICAGO, ILL.
68: 7497

AMERICAN MEDICAL ASSOCIATION,
CHICAGO, ILL. COUNCIL ON MEDICAL
EDUCATION
65: 5997
67: 5999
6000

AMERICAN NATIONAL RED CROSS, NEW
YORK, N.Y.
66: 8294

AMERICAN NATIONAL RED CROSS, NEW
YORK, N.Y. NURSING SERVICES
66: 8295

AMERICAN NURSERYMAN, CHICAGO,
ILL.
65: 401

AMERICAN NURSES' ASSOCIATION, NEW
YORK, N.Y.
65: 4914, 4915
66: 7326
67: 7327
68: 9033

AMERICAN NURSING HOME
ASSOCIATION, WASHINGTON, D.C.
nd: 8321

AMERICAN OCCUPATIONAL THERAPY
ASSOCIATION, NEW YORK, N.Y.
67: 3216
68: 8132

AMERICAN PERSONNEL AND GUIDANCE
ASSOCIATION, WASHINGTON, D.C.
66: 5436
67: 7493

AMERICAN PHILOSOPHICAL SOCIETY,
PHILADELPHIA, PA.
67: 5475

AMERICAN PHYSICAL THERAPY
ASSOCIATION, NEW YORK, N.Y.
67: 6688
68: 6693

AMERICAN REHABILITATION
COUNSELING ASSOCIATION
66: 5436

AMERICAN SOCIETY FOR ENGINEERING
EDUCATION, WASHINGTON, D.C.
62: 4380

AMERICAN SOCIETY FOR PERSONNEL
ADMINISTRATION
67: 7343

AMERICAN SOCIETY FOR TRAINING AND
DEVELOPMENT, MADISON, WIS.
68: 6940

AMERICAN SOCIETY OF
ANESTHESIOLOGISTS, PARK RIDGE,
68: 7497

AMERICAN SOCIETY OF
ANESTHESIOLOGISTS, PARK RIDGE,
ILL.
68: 7497

AMERICAN SOCIETY OF
ANESTHESIOLOGISTS, PARK RIDGE,
ILL.AMERICA
68: 7497

AMERICAN SOCIETY OF CLINICAL
PATHOLOGISTS, CHICAGO, ILL.
67: 5999
6000

AMERICAN SOCIETY OF PLANNING
OFFICIALS
68: 9944

AMERICAN SOCIETY OF RADIOLOGIC
TECHNOLOGISTS, FOND DU LAC, WIS.
nd: 7337
67: 8056

AMERICAN SOCIETY OF RADIOLOGIC
TECHNOLOGISTS, FOND DU LAC, WIS.
EDUCATION COMMITTEE
64: 5840

AMERICAN SOCIETY OF TOOL AND
MANUFACTURING ENGINEERS,
DEARBORN, MICH.
68: 8762

AMERICAN TECHNICAL SOCIETY,
CHICAGO, ILL.
69: 9170, 9494

AMERICAN TELEPHONE AND TELEGRAPH
CO., NEW YORK, N.Y.
65: 4682

AMERICAN VOCATIONAL ASSOCIATION
INC., WASHINGTON, D.C.
69: 8394

AMERICAN VOCATIONAL ASSOCIATION,
INC., WASHINGTON, D.C.
68: 8027
69: 10028

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C.
66: 1827
67: 7493
68: 928
7180, 7380, 7414
8026, 8037
69: 9155, 9654, 9748, 9979
10450, 10452
70: 10317

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. AGRICULTURAL
DIV.
66: 7699

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. AGRICULTURAL
EDUCATION DIV.
69: 10454, 10455

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. BUSINESS AND
OFFICE EDUCATION DIV.
69: 10440

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. COMMITTEE ON
PUBLICATIONS
64: 2496

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. COMMITTEE ON
RESEARCH IN AGRICULTURAL
EDUCATION
63: 2213
68: 8602

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. CURRICULUM
MATERIALS COMMITTEE
70: 10509

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. DISTRIBUTIVE
EDUCATION DIV.
69: 10432

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. HEALTH
OCCUPATIONS SECTION
69: 10434

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. HOME ECONOMICS
EDUCATION DIV.
69: 10453

AMERICAN VOCATIONAL ASSOCIATION,
WASHINGTON, D.C. INDUSTRIAL ARTS
EDUCATION DIV.
69: 10392, 10393, 10394, 10415,

ANNUAL SUPPORTING SERVICE INDEX

VOCATIONAL AND SUPPORTING SERVICES INDEX

AGRICULTURAL EDUCATION
nd: 276, 278
5854
7403, 7405, 7406, 7415,
7417, 7433, 7601, 7843,
7848
8045, 8250
9173, 9282, 9293, 9294,
9298, 9301, 9302, 9309,
9312, 9315, 9318, 9319,
9320, 9321, 9322, 9323,
9324, 9325, 9327, 9328,
9329, 9335, 9338, 9350,
9834
10424
60: 4618
63: 575, 687
1400, 1576
2123, 2213
5271
6502
64: 358, 361, 515, 517,
518, 521, 535, 610,
820, 945
1015, 1443
2309, 2543
3577
4033, 4536, 4538, 4539,
4544, 4545, 4546, 4722,
4723, 4724
6163, 6347
7911
9071, 9078, 9079, 9326
65: 57, 58, 401, 526,
529, 570
1009, 1151, 1373, 1444
2111, 2369
3538
4555, 4562, 4575, 4727
6399, 6426, 6428
8061
9035, 9051, 9252, 9272,
9308, 9310, 9313, 9337,
9341, 9342
66: 372, 373, 374, 375,
376, 377, 383
1131, 1665, 1819, 1949,
1953
3862
4576, 4583, 4584, 4585,
4586, 4840
5186, 5758
6356, 6798, 6823
7165, 7309, 7699
8799
9296, 9300, 9303, 9307,
9314, 9775
67: 475, 971
1457, 1959
2215, 2605, 2609, 2693,
2937
3779
4351, 4644, 4782, 4783,
4784, 4785, 4786, 4787,
4906, 4931
5185
6281, 6282
7160, 7328, 7329, 7416
8603, 8627, 8749
9333, 9336, 9669, 9829,
9843
68: 928
5152, 5545, 5855
6014, 6278, 6336, 6337,
6571, 6725, 6913
7078, 7096, 7181, 7199,
7202, 7284, 7336, 7338,
7351, 7479, 7542, 7557,
7559, 7596, 7597, 7613,
7618, 7619, 7620, 7621,
7628, 7647, 7696, 7698,
7724, 7756, 7809, 7902,
7903, 7930, 7980, 7981
8042, 8074, 8079, 8220,
8252, 8272, 8336, 8602,
8643, 8646, 8656, 8663,
8764, 8766, 8767, 8771,

AGRICULTURAL EDUCATION (CONTINUED)
68: 8822, 8933, 8934, 8935,
8936, 8958, 8984
9077, 9080, 9177, 9287,
9297, 9299, 9311, 9317,
9334, 9339, 9340, 9351,
9772, 9860, 9861, 9999
10000, 10002, 10003, 10004,
10005, 10006, 10007, 10008,
10044, 10045, 10046, 10047,
10048
69: 7965, 7966, 7993
8076, 8299, 8318, 8642,
8765, 8783, 8827, 8828,
8949
9195, 9390, 9392, 9565,
9566, 9567, 9568, 9569,
9610, 9619, 9620, 9621,
9625, 9629, 9630, 9634,
9660, 9668, 9705, 9767,
9778, 9806, 9844, 9858,
9859, 9863, 9871, 9872,
9889, 9898, 9949, 9950,
9956, 9957, 9981, 9982
10001, 10009, 10028, 10035,
10049, 10098, 10116, 10230,
10295, 10302, 10322, 10328,
10370, 10454, 10455, 10456

BUSINESS AND OFFICE EDUCATION
nd: 6189, 6448
7176
8134, 8910
63: 831
3691
6222, 6670
8482
9174
64: 926
1971
5440, 5590, 5821
6745
7205
8658, 8706, 8992
9047
65: 133, 671, 882
1348, 1512, 1599
4028, 4267
9091, 9229, 9251, 9258,
9271
66: 444
2064, 2367
6190, 6396, 6397
7193
8130
9250, 9967, 9968
67: 1577
2234, 2706, 2739
3598, 3640
5917
7032, 7044, 7067
8249, 8840
9191
68: 5935
6720, 6731, 6971
7184, 7186, 7235, 7286,
7432, 7435, 7500, 7561,
7605, 7624, 7653, 7692,
7733, 7777, 7786, 7975,
7976, 7977, 7978
8024, 8057, 8374, 8823,
8839
9053, 9363, 9364, 9365,
9498
69: 8540, 8747, 8913, 8923,
8937, 8939, 8940, 8941,
8942, 8943
9488, 9670, 9673, 9977
10440

DISTRIBUTIVE EDUCATION
nd: 7519
8060, 8069, 8095, 8821,
8824, 8944
9295, 9557
50: 124
52: 123, 152, 173

DISTRIBUTIVE EDUCATION (CONTINUED)
56: 125
57: 126
62: 146, 155, 156, 170
63: 150
2375
8484, 8583
64: 163, 165, 166, 167
1390
2265
3608
65: 209, 210, 215, 423
2898
4682
7690
66: 1621
2493
7518
9270
67: 2617
5973, 5974, 5975, 5976
6011, 6746, 6810
7217
8296, 8599, 8772
10056
68: 7014, 7137, 7246, 7420,
7616, 7697, 7766, 7851,
7952
8021, 8094, 8191, 8195,
8245, 8478, 8479, 8773,
8885, 8938, 8989
9361, 9599
69: 7953
8453, 8454, 8991
9503, 9805, 9877, 9880,
9897, 9951
10401, 10432

HEALTH OCCUPATIONS EDUCATION
nd: 6028, 6145, 6863
7136, 7191, 7337, 7969
8283, 8321, 8418, 8931,
8948
63: 1254
2028, 2114, 2446
4350, 4357
5968
8825
64: 1086, 1532, 1900
4044, 4302, 4339
5149, 5560, 5574, 5840
8616
9090
65: 365, 590
2541, 2555, 2557, 2558,
2599
3221, 3684, 3972
4643, 4652, 4914, 4915
5214, 5358, 5646, 5997
6027, 6127
8763
9036, 9678
66: 1891, 1964
2143, 2393, 2638, 2920
3619, 3641, 3812
4251, 4358, 4556, 4623,
4801
5153, 5600, 5641, 5822
6208, 6254, 6348, 6349,
6413, 6658, 6772
7109, 7326, 7556, 7855
8063, 8162, 8293, 8376,
8461, 8654
9267
67: 192, 771, 818
1078
2138, 2216, 2691, 2888
3216, 3218, 3501, 3615
4085, 4173, 4179, 4240,
4247, 4356, 4624, 4706,
4789, 4790, 4825
5162, 5244, 5316, 5425,
5426, 5427, 5476, 5635,
5954, 5998, 5999
6000, 6314, 6657, 6688,
6697, 6698
7046, 7062, 7091, 7327,

MANPOWER ECONOMICS (CONTINUED)
69: 8780
9245, 9256, 9519, 9759,
9798, 9936
10065, 10099, 10180, 10247,
10414, 10450

OCCUPATIONAL PSYCHOLOGY
64: 5825
65: 2855
66: 2700
3583
7106
67: 4094
7045, 7110
69: 8507

OCCUPATIONAL SOCIOLOGY
nd: 7778
64: 2145
4720
65: 3620
8240
66: 3745, 3756, 3757, 3845,
3860
5225, 5226
6424
7854
67: 3754, 3755, 3758, 3760,
3863
5475
6564, 6624, 6968
7033, 7061, 7066, 7098
68: 5631
6825, 6846, 6862
7048, 7081, 7103, 7146,
7215
8105

TECHNICAL EDUCATION
nd: 5857
6145
10029, 10033, 10034
62: 4380, 4420
63: 4015
64: 515, 592
1559
4518
5270
65: 591
1476, 1524, 1973, 1974,
1988
2301
3857
10443
66: 1942, 1975
2692
3641
4319
8414, 8999
9331
67: 1409
2554
4233, 4792
5610, 5977, 5978
8096, 8334, 8972, 8988
9344
68: 4791
5467, 5628
6858, 6874, 6912
7247, 7414, 7419, 7497,
7561, 7627, 7694, 7756,
7871
8001, 8151, 8176, 8272,
8402, 8486, 8656, 8669,
8870, 8983
9343, 9362, 9379, 9413
69: 7875, 7876, 7961, 7989,
7993
8165, 8318, 8431, 8470,
8472, 8556, 8722, 8786
9163, 9254, 9260, 9370,
9377, 9490, 9526, 9582,
9588, 9603, 9767
10035, 10139, 10140, 10141,
10150, 10151, 10152, 10307,
10435, 10444

TRADE AND INDUSTRIAL EDUCATION
nd: 2848

TRADE AND INDUSTRIAL EDUCATION (CONTINUED)
nd: 5579, 5857
7984
8192, 8194, 8291, 8332,
8527, 8553, 8570, 8610,
8613, 8617, 8626, 8805,
8808, 8809, 8810, 8811,
8812, 8911
9284, 9289, 9345, 9615,
9616, 9617, 9756, 9801,
9926
10029, 10032, 10033, 10034,
10037
58: 7394
59: 9592
60: 7396
62: 717
4192, 4380, 4420
6480
7395
63: 2338, 2383
4015, 4800
8806
10031, 10038
64: 432, 592
1339, 1678, 1835
2125, 2297
3189, 3808
4211, 4518, 4611, 4637
5270, 5617, 5717
8154, 8915
9925
10055, 10088, 10089, 10091
65: 591, 861
1619, 1916, 1974
2100, 2397, 2398, 2400,
2401, 2874
3901
4408, 4508, 4600, 4798,
4947
7397
8287, 8288, 8289, 8625,
8959, 8960, 8961, 8985,
8986
9001, 9002, 9003, 9023,
9024, 9025, 9026, 9027,
9028, 9029, 9030, 9034,
9052, 9230, 9305, 9415,
9711, 9737, 9739
66: 1288, 1367, 1552, 1643,
1718, 1942, 1950, 1961,
1975
2593, 2692, 2763
3280, 3554
4799
6484, 6568
7398, 7399, 7400, 7496
8136, 8290, 8414, 8415,
8611, 8624, 8651, 8668,
8857
9331, 9718, 9719, 9720,
9721, 9722, 9723, 9724,
9725, 9726, 9727, 9728,
9729, 9730, 9731, 9732,
9733, 9734, 9735, 9736,
9738, 9740, 9741, 9742,
9743, 9744, 9745, 9746,
9747, 9966
10019
67: 1284, 1409
2214, 2554, 2676, 2678,
2820
3533, 3595
4264, 4392, 4393, 4712,
4773, 4803, 4946
5470, 5597, 5610, 5623,
5640, 5978
6123, 6804
7084, 7101, 7390, 7391,
7392, 7484, 7485
8096, 8334, 8388, 8535,
8607, 3841, 8988
9344, 9374, 9672, 9692,
9707, 9708, 9854
68: 4772, 4791, 4907
5438, 5467, 5639
6194, 6371, 6417, 6519,
6581, 6676, 6678, 6681,
6682, 6739, 6740, 6818,

TRADE AND INDUSTRIAL EDUCATION (CONTINUED)
68: 6838, 6858, 6859, 6874,
6916, 6917
7171, 7185, 7258, 7322,
7349, 7361, 7393, 7461,
7465, 7487, 7504, 7507,
7540, 7584, 7590, 7607,
7629, 7694, 7723, 7752,
7768, 7870, 7871, 7874,
7913
8026, 8027, 8176, 8196,
8312, 8337, 8338, 8339,
8340, 8341, 8342, 8343,
8344, 8345, 8346, 8347,
8348, 8349, 8350, 8351,
8352, 8353, 8354, 8355,
8356, 8357, 8358, 8359,
8360, 8361, 8362, 8402,
8447, 8505, 8569, 8577,
8584, 8623, 8633, 8676,
8714, 8724, 8731, 8838,
8870, 8907, 8983
9022, 9180, 9330, 9343,
9362, 9366, 9367, 9368,
9369, 9371, 9372, 9373,
9375, 9376, 9378, 9379,
9380, 9413, 9671, 9675,
9693, 9709, 9710
69: 7875, 7876, 7961, 7989,
7991, 7998
8271, 8319, 8470, 8472,
8494, 8516, 8536, 8648,
8652, 8666, 8683, 8722,
8925
9176, 9254, 9260, 9277,
9370, 9377, 9582, 9588,
9590, 9814, 9839, 9869,
9890, 9900
10139, 10140, 10141, 10150,
10151, 10152, 10304, 10305,
10431

GENERAL VOCATIONAL AND TECHNICAL EDUCATION
nd: 1463
2641
4352
5483
6143, 6781, 6816
7273, 7295, 7362, 7410,
7661, 7701, 7918
8244, 8300, 8432, 8526,
8573, 8701, 8739, 8849
9395
56: 5637
59: 5256
60: 2135
62: 2839
63: 95, 190
1489, 1669, 1702, 1967
2870, 2947
4035, 4121, 4837, 4950
5178, 5295, 5604
6402
8009, 8442
64: 569
1368, 1411, 1442, 1474,
1553, 1555, 1559, 1624,
1687, 1688, 1722
2496, 2909
3219, 3287, 3288, 3289,
3290, 3524, 3584, 3613,
3826
4275, 4535, 4675, 4679
5603, 5615
6168
8680
65: 477
1029, 1161, 1394, 1436,
1476, 1524, 1606, 1679,
1695, 1728, 1797, 1973,
1988
2006, 2010, 2103, 2104,
2109, 2132, 2191, 2192,
2193, 2194, 2236, 2301,
2359, 2511, 2564, 2574,
2651, 2823, 2854
3685, 3857
4277, 4371, 4424, 4728,

ANNUAL DOCUMENT NUMBER INDEX

VT NUMBER, ISSUE, AND PAGE NUMBER INDEX

VT NO	ISSUE	PAGE
49	ASU70	484
55	SP70	488
57	AF 69	2
58	AF 69	2
89	ASU70	496
94	ASU70	496
95	ASP70	382
98	ASU70	496
107	ASU70	496
117	ASP70	348
118	ASU70	497
120	ASU70	497
121	ASU70	497
123	AF 69	22
124	AF 69	22
125	ASP70	340
126	ASP70	340
133	ASP70	336
135	ASU70	484
146	AF 69	22
150	AF 69	22
152	AF 69	22
155	AF 69	23
156	AF 69	23
163	AF 69	23
165	ASP70	340
166	AF 69	23
167	AF 69	24
170	AF 69	24
173	AF 69	24
176	ASU70	497
190	F 69	38
192	W 69	242
209	AF 69	25
210	AF 69	25
215	AF 69	25
275	ASU70	464
276	ASP70	316
277	ASU70	464
278	ASP70	316
279	ASU70	464
280	ASU70	465
281	ASU70	465
333	ASP70	356
338	ASP70	348
339	ASU70	498
349	ASP70	348
352	ASU70	498
358	ASP70	316
361	ASP70	316
365	W 69	284
367	ASU70	465
368	ASU70	466
372	ASP70	317
373	ASP70	317
374	ASP70	317
375	ASP70	317
376	ASP70	317
377	ASP70	318
383	ASP70	318
385	ASP70	348
386	ASU70	498
387	ASU70	499
388	ASP70	348
389	ASU70	499
392	ASP70	349
398	ASP70	356
401	ASP70	318
404	AF 69	38
423	ASP70	340

VT NO	ISSUE	PAGE
432	AF 69	54
444	ASP70	336
445	ASU70	499
446	ASU70	500
447	ASU70	500
458	F 69	2
467	ASP70	356
475	AF 69	2
477	SU70	784
506	ASP70	356
515	ASP70	318
517	ASP70	318
518	ASP70	318
521	ASP70	319
526	ASP70	319
528	ASU70	466
529	W 69	242
535	W 69	232
544	ASU70	466
569	ASP70	382
570	ASP70	319
575	AF 69	2
587	W 69	192
590	ASP70	344
591	ASP70	362
592	ASP70	362
610	ASP70	319
614	F 69	94
671	ASP70	336
679	SP70	442
687	ASP70	320
717	AW 69	212
752	ASU70	524
771	SP70	442
774	F 69	38
816	SP70	560
818	SP70	488
820	AF 69	2
831	ASP70	336
842	AF 69	38
861	AF 69	54
875	ASU70	500
877	W 69	192
882	ASP70	336
888	ASU70	501
889	ASU70	501
890	ASP70	349
891	ASU70	501
892	ASU70	502
893	ASU70	502
898	AF 69	48
926	ASP70	337
928	F 69	94
945	ASP70	320
967	SP70	442
970	SP70	488
971	SP70	570
979	ASP70	356
980	ASP70	357
1009	AF 69	3
1015	ASP70	320
1029	SP70	478
1032	ASP70	349
1036	SP70	478
1078	SP70	442
1086	F 69	38
1087	ASU70	480
1088	ASP70	349
1126	W 69	242
1131	ASP70	320

VT NO	ISSUE	PAGE
1151	ASP70	320
1161	AF 69	72
1171	W 69	192
1200	ASP70	357
1211	W 69	192
1231	SU70	792
1254	SP70	478
1258	ASP70	357
1284	AF 69	54
1288	ASP70	362
1289	W 69	300
1315	ASU70	480
1319	SP70	443
1325	ASP70	350
1339	W 69	232
1343	SU70	766
1345	SP70	489
1348	AF 69	14
1367	SP70	489
1368	SP70	560
1373	W 69	242
1390	W 69	193
1394	W 69	232
1395	W 69	232
1396	W 69	233
1397	W 69	233
1400	ASP70	321
1409	W 69	318
1411	SP70	443
1421	W 69	193
1432	ASU70	467
1436	ASP70	382
1441	SP70	489
1442	F 69	76
1443	W 69	338
1444	W 69	338
1449	W 69	193
1455	ASU70	548
1457	W 69	338
1463	SP70	443
1474	W 69	193
1476	SP70	532
1481	SP70	489
1489	W 69	194
1509	SP70	443
1512	W 69	243
1514	SP70	560
1521	F 69	2
1524	W 69	332
1532	F 69	2
1535	AF 69	38
1552	F 69	2
1553	F 69	38
1555	W 69	284
1559	F 69	2
1561	SP70	444
1572	F 69	39
1576	SP70	560
1577	AW 69	182
1599	F 69	3
1606	W 69	194
1614	SP70	548
1619	W 69	194
1621	SP70	444
1624	SU70	746
1643	W 69	318
1646	ASU70	502
1665	AF 69	3
1669	SU70	746
1673	W 69	194

VT NUMBER, ISSUE, AND PAGE NUMBER INDEX

VT NO	ISSUE	PAGE
2880	F 69	88
2886	F 69	15
2888	SU70	748
2898	F 69	104
2909	F 69	16
2916	F 69	16
2920	F 69	16
2937	AW 69	162
2947	F 69	16
3189	AF 69	55
3216	W 69	196
3218	W 69	246
3219	F 69	41
3221	F 69	41
3229	F 69	17
3234	W 69	246
3245	F 69	17
3280	F 69	17
3287	F 69	42
3288	F 69	42
3289	F 69	42
3290	F 69	42
3291	W 69	246
3293	W 69	247
3312	F 69	76
3314	F 69	42
3362	W 69	247
3363	ASP70	382
3372	W 69	247
3409	F 69	43
3433	SU70	766
3435	W 69	248
3496	F 69	43
3501	SP70	447
3505	F 69	76
3521	AF 69	38
3523	F 69	18
3524	W 69	285
3533	AW 69	213
3538	F 69	43
3554	F 69	18
3570	SU70	766
3577	W 69	248
3583	W 69	248
3584	F 69	18
3587	ASU70	503
3595	SU70	767
3598	AW 69	182
3608	AF 69	26
3613	F 69	18
3615	SP70	552
3619	F 69	19
3620	W 69	249
3640	W 69	197
3641	W 69	197
3648	F 69	44
3649	F 69	44
3650	F 69	44
3652	F 69	45
3668	F 69	45
3669	W 69	249
3684	F 69	45
3685	W 69	197
3691	W 69	198
3693	W 69	249
3697	F 69	45
3700	F 69	19
3719	W 69	249
3723	SU70	793
3745	W 69	250
3746	F 69	46
3748	F 69	46
3751	W 69	250
3752	F 69	46
3753	F 69	47
3754	F 69	47
3755	F 69	77
3756	F 69	47
3757	F 69	77
3758	F 69	77
3760	W 69	250
3773	F 69	19
3779	W 69	326
3782	F 69	47
3796	AF 69	72
3800	W 69	250
3801	W 69	251
3802	W 69	251
3808	W 69	251
3812	W 69	252
3814	W 69	252
3817	W 69	252
3826	W 69	252
3833	F 69	48
3834	F 69	48
3835	W 69	253
3836	F 69	48
3837	F 69	48
3838	F 69	49
3839	F 69	49
3840	F 69	49
3841	F 69	49
3842	W 69	253
3843	F 69	50
3844	W 69	253
3845	W 69	254
3846	W 69	254
3849	F 69	50
3850	W 69	254
3852	W 69	254
3857	F 69	50
3860	W 69	255
3862	F 69	51
3863	W 69	255
3868	F 69	77
3873	W 69	300
3875	F 69	51
3876	F 69	51
3882	W 69	255
3883	W 69	256
3893	F 69	51
3900	SU70	767
3901	SU70	793
3902	SU70	793
3903	W 69	256
3942	SP70	570
3960	F 69	52
3961	F 69	52
3962	F 69	52
3972	SP70	495
3974	SU70	767
4015	SU70	762
4019	W 69	332
4026	W 69	198
4028	AW 69	182
4033	AW 69	162
4035	W 69	198
4044	W 69	198
4085	SP70	532
4094	W 69	256
4121	F 69	20
4124	F 69	52
4125	F 69	53
41[illegible]0	ASU70	524
4173	W 69	199
4179	F 69	53
4187	W 69	296
4192	AF 69	55
4196	W 69	199
4211	F 69	104
4233	W 69	199
4240	W 69	256
4247	F 69	20
4249	F 69	53
4251	F 69	20
4264	AF 69	55
4267	W 69	199
4275	F 69	20
4277	W 69	322
4301	F 69	84
4302	W 69	339
4319	F 69	20
4322	F 69	53
4323	F 69	54
4339	W 69	339
4341	W 69	200
4342	F 69	21
4350	F 69	21
4351	W 69	257
4352	SU70	832
4353	SU70	784
4356	W 69	257
4357	F 69	21
4358	F 69	22
4359	F 69	54
4360	F 69	22
4371	F 69	54
4372	W 69	301
4375	F 69	54
4380	ASP70	363
4392	F 69	84
4393	SU70	748
4401	SU70	748
4408	W 69	257
4419	W 69	235
4420	F 69	55
4424	W 69	257
4503	F 69	55
4508	W 69	200
4509	F 69	55
4510	F 69	55
4518	W 69	200
4519	W 69	201
4521	F 69	56
4526	W 69	201
4532	W 69	201
4535	F 69	112
4536	F 69	112
4538	W 69	201
4539	W 69	258
4544	W 69	258
4545	W 69	202
4546	W 69	285
4555	W 69	258
4556	F 69	78
4558	ASU70	467
4562	W 69	318
4563	W 69	259
4570	F 69	56
4572	F 69	58